1986

Development of flight performance algorithms and a tactical computer aided mission planning system for the A-7E aircraft.

Nutter, Christopher G.

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

DEVELOPMENT OF FLIGHT PERFORMANCE ALGORITHMS AND A TACTICAL COMPUTER AIDED MISSION PLANNING SYSTEM FOR THE A-7E AIRCRAFT

by

Christopher Glenn Nutter

September 1986

Thesis Advisor: D. M. Layton

Approved for public release; distribution is unlimited

URITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b. RESTRICTIVE MARKINGS
SECURITY CLASSIFICATION AUTHORITY	3 DISTRIBUTION / AVAILABILITY OF REPORT Approved for public release; distribution is unlimited.
DECLASSIFICATION / DOWNGRADING SCHEDULE	
PERFORMING ORGANIZATION REPORT NUMBER(S)	S MONITORING ORGANIZATION REPORT NUMBER(S)
NAME OF PERFORMING ORGANIZATION val Postgraduate School	6b OFFICE SYMBOL (If applicable) 67
7a NAME OF MONITORING ORGANIZATION Naval Postgraduate School	
ADDRESS (City, State, and ZIP Code) onterey, California 93943-5000	7b. ADDRESS (City, State, and ZIP Code) Monterey, California 93943-5000
NAME OF FUNDING / SPONSORING ORGANIZATION	8b. OFFICE SYMBOL (If applicable)
9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER	
ADDRESS (City, State, and ZIP Code)	10 SOURCE OF FUNDING NUMBERS

PROGRAM ELEMENT NO	PROJECT NO	TASK NO	WORK UNIT ACCESSION NO

TITLE (Include Security Classification)
/ELOPMENT OF FLIGHT PERFORMANCE ALGORITHMS AND A TACTICAL COMPUTER AIDED SSION PLANNING S STEM FOR THE A-7E AIRCRAFT.

PERSONAL AUTHOR(S)
tter, Christopher, G.

TYPE OF REPORT	13b TIME COVERED	14 DATE OF REPORT (Year, Month, Day)	15 PAGE COUNT
ster's Thesis	FROM _____ TO _____	1986 September	122

SUPPLEMENTARY NOTATION

COSATI CODES		
FIELD	GROUP	SUB-GROUP

18 SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
Aircraft Performance, A-7E Aircraft, Mission Planning, Computerized Mission Planning, IBM-PC.

ABSTRACT (Continue on reverse if necessary and identify by block number)

This thesis presents a fully developed, tactically oriented computer aided mission planning system for the A-7E aircraft. The system is designed to be extremely easy to operate by someone with no computer experience, and is a replacement for 53 NATOPS performance charts that are most applicable to flight and mission planning. High altitude performance, as well as low altitude and maximum range performance, are available. Tactical navigation routes may be entered, edited, and saved in a disk file. Navigation computations are linked with aircraft performance to provide printout of a completed mission planning jet log.

DISTRIBUTION / AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION unclassified
NAME OF RESPONSIBLE INDIVIDUAL Donald M. Layton	22b TELEPHONE (Include Area Code) (408) 646-2491 / 22c OFFICE SYMBOL 67 LN

FORM 1473, 84 MAR 83 APR edition may be used until exhausted
All other editions are obsolete

SECURITY CLASSIFICATION OF THIS PAGE

BLOCK NUMBER 19 (continued):

Numerical techniques for obtaining analytical expressions from chart and tabular data included multiple linear regression analysis, curve fitting, and cross plotting of regression coefficients. The computer program results have been correlated with NATOPS data and actual flight test, and have been found to be highly accurate. The program is designed to be run on the IBM PC/XT/AT and compatible computers with a minimum of 256K memory available.

Approved for public release: distribution is unlimited.

Development Of Flight Performance Algorithms And A Tactical Computer Aided Mission Planning System For The A-7E Aircraft

by

Christopher G. Nutter
Lieutenant, United States Navy
B.S., Purdue University, 1976
M.A., Webster College, 1980

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN AERONAUTICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
September 1986

ABSTRACT

This thesis presents a fully developed, tactically oriented computer aided mission planning system for the A-7E aircraft. The system is designed to be extremely easy to operate by someone with no computer experience, and is a replacement for 53 NATOPS performance charts that are most applicable to flight and mission planning. High altitude performance, as well as low altitude and maximum range performance, are available. Tactical navigation routes may be entered, edited, and saved in a disk file. Navigation computations are linked with aircraft performance to provide printout of a completed mission planning jet log.

Numerical techniques for obtaining analytical expressions from chart and tabular data included multiple linear regression analysis, curve fitting, and cross plotting of regression coefficients. The computer program results have been correlated with NATOPS data and actual flight test, and have been found to be highly accurate. The program is designed to be run on the IBM PC/XT/AT and compatible computers with a minimum of 256K memory available.

THESIS DISCLAIMER

Neither the U.S. Navy nor anyone else who has been involved in the creation, production, or delivery of this program shall be liable for any direct, indirect, consequential, or incidental damages arising out of the use, the results of use, or inability to use such product. Complete and thorough preflight and tactical planning is the responsibility of the officer in charge of such activities, and should be accomplished using accepted techniques and the appropriate officially approved publications. This program is a planning aid and all computed solutions should be carefully validated.

TABLE OF CONTENTS

LIST OF FIGURES

ACKNOWLEDGMENT

My thanks to Professor D.M. Layton for his professional assistance and support in the pursuit of this thesis. My thanks also to CDR. W.M. Siegel, USN (Ret), and to CDR. R.G. Bettinger, USN, for their professional leadership and support.

I. INTRODUCTION

The objective of this thesis was to produce a fully developed and correlated, computer based mission planning system. Current mission planning computations are done by hand calculation with aircraft performance data derived from graphical interpretation of NATOPS chart presentations. That method is slow and generates solutions which may contain significant errors. A class of sixteen prospective squadron commanding officers attending a one-week Command Safety Course at the School of Aviation Safety, Naval Postgraduate School, Monterey, California, was asked to complete a typical mission planning problem . The results were that the correct answer was not achieved by any member of the class. The range of solutions was 336 nm to 868 nm, spread about the correct answer of 538 nm. The correct answer was achieved only by the class instructor after a measured 16 hours of effort with the NATOPS manual [Ref. 1].

A study by the Vought Corporation to evaluate the feasibility of a computer based flight planning system resulted in a recommendation to the Naval Air Developement Center (NADC), Warminster, Pennsylvania, that such a system would be feasible and would result in fuel savings as well as increased accuracy in the mission planning

areas [Ref. 2]. In addition, Hill [Ref. 3] clearly defines the need for accurate aircraft performance data to be available in order for strike planners to plan safe and tactically effective long range missions of 800 to 1200 nm radius. These missions are complex evolutions and may be significantly compromised by a false need for additional support aircraft such as electronic countermeasures, radar surveillance, and tanker assets. A recent article [Ref. 4:pp.69-73], detailing a report on the differences between Navy and Air Force aviation operations was completed in May 1985 as a result of a study directed by the Vice Chief of Naval Operations on this subject. This report directly addressed a major difference regarding aircrew duties and aircrew fatigue:

> The familiar complaint of excessive collateral duties was of major interest to the exchange pilots. There was unanimous agreement that in the Navy these non-flying tasks impede the development of flying skills, particularly in the junior ranks, while contributing to fatigue, hasty flight planning, and sketchy briefings and debriefings. In the Air Force, collateral duties are directly related to flying and help fulfill requirements for professional development of the officer and the aviator. . . .Navy aircrews devote less time to flight planning than Air Force fliers. Although briefings and post-flight debriefings are valued in both services, Navy briefings were considered non-standard when compared to those in the Air Force. Again, collateral duty responsibilities were believed to detract from better fight planning and briefing in the Navy. The Air Force is more strictly regulated than the Navy. The Navy system fosters the bending of rules to fit operational commitments, a practice which compromises safety, and promotes a side effect--tacit approval of nonadherence to regulations.
>
> NATOPS was created in 1961 and is credited with being a milestone in naval aviation safety improvement. Yet, the study group found that NATOPS is less effective

in promoting safety than the Air Force's Standardization and Evaluation (Stan/Eval) program. Overall, the NATOPS program is sound, but enforcement of the procedures is occassionally weak. Furthermore, the Air Force's Eval checks are more stringent than the Navy's NATOPS checks.

II. APPROACH TO THE PROBLEM

A. BACKGROUND

Development of a computer based mission planning system required examining constraints in hardware/software performance and availability, and in deriving the necessary mathematical equations and algorithms. Until 1984, the primary consideration in developement of a computer based system was the high cost and limited performance of hardware. Since 1984, these problems have been eliminated due to the advance in micro computer technology. Central processing units have tremendous speed and capability, and additional memory chips have become so inexpensive that one megabyte of Random Access Memory (RAM) is common on many personal computers. As a result of application programs written for users with a broad range of technical backgrounds, and development of very powerful languages, software for personal computers has become, in may ways, more usable than main frame programs. Many main frame capabilities now reside also with the personal computer.

Several attempts at deriving adequate mathematical equations that represent the NATOPS curves have resulted with only partial success. Koger [Ref. 5] employed an extension of the technique developed by Siegel [Ref. 1] to derive mathematical expressions through a least squares

fit modeling method for takeoff, maximum range, maximum endurance, and low level aircraft performance. These equations, however, do not fully describe the entire NATOPS chart for which they were developed. Koger [Ref. 5] explained the difficulties encountered with charts containing different families of curves, non-linear coefficients required in regression analysis, and regression of curves that were not well behaved. Consequently, these equations may only be accurately applied over a limited range of mission parameters. Hill [Ref. 3] developed equations and software for computing aircraft performance during the climb, cruise, and descent mission phases. One of the two primary limitations to Hill's program is its reliance on some equations applicable only to limited areas of the chart for which they were intended to describe. A second primary limitation is the program's dependance on published NATOPS drag count data, computed as a function of very specific aircraft loading information which must be looked up and supplied by the user. Obtaining this input data is a time intensive exercise, the results of which are frequently not completely accurate. Recent investigation of NATOPS drag count data by Naval Weapons Center, China Lake, and the Vought Corporation has revealed an underprediction of the actual drag counts by 25 to 30 counts, or approximately 25% error in total drag. Actual aircraft

installations with flight test data show higher drag and fuel requirements than those computed from NATOPS [Ref. 6].

In support of a contract between NADC and Vought Corporation to develop the Flight Performance Advisory System (FPAS) for Navy A-7E aircraft, Vought undertook a two year program, from December 1980 until January 1982, designed to develop flight performance algorithms and equations which could be programmed into the TC-2A Navigation and Weapons Delivery Computer (NWDC) on-board the A-7E aircraft [Ref. 7]. The study utilized actual aerodynamic and propulsion data bases, accounted for nonstandard atmospheric conditions, included algorithms for computation of cockpit instrument readings (indicated altitude, airspeed, and mach number), and allowed computations for partial power climbs and descents. Results from the generated equations and algorithms were correlated and verified within 1% of those from a more precise program running on the Cyber 175 computer. This study did not evaluate takeoff performance.

As a foundation for the mission planning system, accurate equations were required which described the full range of each applicable NATOPS chart for all mission areas. These mission planning areas were the takeoff, climb, high and low level cruise, descent, maximum range profile at maximum range altitude, maximum range possible

at an alternate altitude, high and low level navigation route computations, and the means to edit and store tactical navigation route structures. Complete aircraft performance data for each mission area, and coupled with the navigation route data to produce completed flight log cards, was the goal of this study.

B. EQUIPMENT

The work in obtaining satisfactory equations from NATOPS data and charts was performed on an IBM PC personal computer using relatively sophisticated multiple regression software [Ref. 8], and on the Hewlett-Packard HP-41 hand calculator using a set of four statistical regression programs written by the author. The mission planning software was developed with the author's equations obtained from statistical regression analysis of 20 NATOPS graphs for the takeoff, climb, and optimum performance mission phases. Each graph contained up to 9 curves and each curve was a function of primarily four independant variables. Additional equations were obtained from statistical regression and aerodynamic analysis by Vought [Ref. 7] which included another 33 NATOPS graphs for the cruise, descent, and optimum performance mission phases.

III. SOLUTION

A. PROCEDURE

The general procedure for statistical analysis of NATOPS graphical data is multiple regression. Each curve on a chart is regressed according to the same family of curves. The coefficients obtained in the equations are plotted against a common parameter of significance present in the initial graph. Equations are obtained to solve for the coefficients as a function of independant variables, which are then inserted into the equation of the family of original curves containing other independant variables. The result is a single equation to compute a dependant variable, and which contains all the other independant variables present in the original plots. This technique was investigated by Siegel [Ref. 1] and is clearly described by Koger [Ref. 5]. The technique is one of many numerical methods that may be applied to the problem of multiple regression of graphical data, however, this technique is not clearly presented in texts and is not a common or popular method of analysis. The requirement for application of computer methods in the analysis has been a factor in its lack of popularity. The multiple regression model used is a standard model and is described in Park [Ref. 8] and Ott/Hildebrand [Ref. 9]. Complete descriptions of this basic model and other techniques are

contained in an excellent text by Draper and Smith [Ref. 10].

Many of the NATOPS charts are plots of a dependant variable as a function of up to four independant variables. This complicates the multiple regression procedures, but is quite manageable. A number of technical difficulties can, however, significantly reduce the effectiveness of this method of analysis. The most prevalent problem is a plot of data in which all curves are not of the same family. This is evident in the maximum refusal speed chart for takeoff factor shown in Figure 1, in which the nature of the plotted data is clearly not consistant or of the same family. In these curves, additional numerical and statistical analysis is applied using computer methods, and a family is found for the curves that adequately describes them entirely. One method is to plot the graphical data using a reversed coordinate system and carefully chosen unit gradations, then to apply the multiple regression procedure to this "new" graph. Occassionally this resulted in a set of original curves more well behaved for analysis and closer to a common family of curves. Simply adding terms to the initial regression model is not satisfactory and, in fact, complicates the statistical analysis by positively influencing computed parameters indicative of how good a

model is. This results in a final set of equations which do not describe the original graphical data.

Obtaining accurate mathematical equations was a basic and preliminary requirement for addressing the problem which was the objective of this thesis. Developement of a fully functional computer aided mission planning program required accurate equations for the entire range of possible aircraft configurations, and the entire range of 53 NATOPS graphical plots.

IV. RESULTS

A. DISCUSSION

The final equations obtained from the author's analysis and the work by Vought [Ref. 7] were used in the programming effort, and are accurate within 1% of NATOPS results for most operational aircraft configurations. At extremes of either environmental or aircraft parameters, the final mission planning program yields results within 2% of NATOPS results. It should be pointed out that most of the NATOPS charts for the A-7E aircraft can not be read to within 2% and that this degree of accuracy over the entire operating range of the aircraft is considered acceptable.

The result of the programming effort is a mission planning system which enables pilots with no experience in computers to obtain complete aircraft performance for all mission phases. Navigation routes may entered, edited, and stored on disk. These may be called later to speed the planning process. High or low navigation route data is combined with the appropriate performance computations, and the system will produce a completed jet log for the high and low mission areas. The experimental planning exercise described by Siegel [Ref. 1] required 16 hours of work with the NATOPS charts to obtain a correct solution. This same problem can be solved using the mission planning

system software developed in this thesis, and requires approximately 30 minutes. This is a decrease in mission planning time of 96.9 percent. All results can be printed manually or automatically, an option selected by the user.

In order to fully describe the software to users, and explain each of the program's features, options, and logic flows, a complete user manual has been written as Appendix B. A feature of this manual is a quick start card which allows users to immediately begin planning without reading the documentation. The program has been written with on-screen help in order to accomodate this user approach. Each data input/output screen is detailed in Appendix B.

B. CORRELATION

Computed numerical results have been compared with four separate sources and have been found to be very accurate to over the entire operating envelope of the aircraft. Program computations were verified with numerical examples in the Vought work [Ref. 7], manual data extraction from the NATOPS charts, performance computations made with the Navy's Optimum Path Aircraft Routing System (OPARS) [Ref. 11], and fight test data obtained from an informal analysis conducted by a U.S. Navy test pilot and Naval Weapons Center, China Lake, California.

V. CONCLUSIONS AND RECOMMENDATIONS

This mission planning system software reduces planning time significantly, and allows missions to be planned with increased safety and mission effectivenenss. A broad range of tactical options can be studied for tradeoffs in mission requirements, and can enable the planner to quickly compute requirements and a jet log for any of the input routes, or any of the routes on file. These tactical options may be required for consideration due to changes in the assigned targets, current intelligence regarding enemy target location or defensive placement of anti-aircraft weapons, or last minute changes in political cooperation or policy which may dictate severe changes to operational planning.

This program will allow pilots and aircrews to spend more time studying the target, briefing the mission, and reviewing complex aircraft emergency procedures, and significantly less time tracing through NATOPS charts for basic aircraft and mission performance data.

A single recommendation is necessary. Computed performance data under a broad range of aircraft configurations and environmental conditions should be verified. This can be most economically accomplished if each A-7E squadron and each functional wing were given a

copy of the program and the user manual, and kept a record of computed performance versus actual performance. The program should also be distributed to Air Force Tactical A-7D units, and Air National Guard A-7D units. Although the Air Force A-7D is lighter than the Navy A-7E, the program is written to accomodate input weights, therefore, computed aircraft performance should be very close to that actually experienced by Air Force and Air National Guard units.

APPENDIX A

TACTICAL COMPUTER AIDED MISSION PLANNING SYSTEM SOURCE CODE

```
0 '   TOPHALF - 5/15/1986 - 22:17:18
1 '*****************************************************
2 'TACTICAL COMPUTER AIDED MISSION PLANNING SYSTEM
3 '  WRITTEN BY LT. CHRIS NUTTER, USN
4 '  AERONAUTICAL ENGINEERING DEPT.
5 '  NAVAL POSTGRADUATE SCHOOL
6 '  MONTEREY, CALIFORNIA
7 '*****************************************************
8 COPYRIGHT.SS$="(C)Copyright 84,85 The Software Bottling
Company Of New York"
10 DIM VL.SS$(100), LO.SS%(100,2), LE.SS%(100),
TY.SS$(100), PIC.SS$(100),RG.SS(100,2), CL.SS%(100,2),
SPECCHR.SS%(100)
11 DIM TODATA(4,3),FUELFLO(18,3),THRUST(9,2),W(9),F(3)
12 DIM CLBDATA(8),DESCH(4),DESDATA(4),A%(3)
14 SCREEN 0,1,3,0:CLS:SCREEN 0,1,2,0:CLS:SCREEN
0,1,1,0:CLS:SCREEN 0,1,0,0:CLS
15 PRINT "~L=NAVDATA/"
16 PRINT "~C=ALL/"
17 TOTALDIST=0
20 KEY OFF: SD.SS%=1 : NUMSCR.SS%=3 : BLNK.SS$=SPACE$(78)
30 GOSUB 62890 'Test For Mono OR Color Display
40 COLOR 15,1,4
45 GOSUB 7700
50 'BRING UP XCINPUT.SCR (XCFORM.BAS) FOR DATA ENTRY
51 SCR.SS%=1:INIT.SS%=-1:GOSUB 60000
55 SCREEN 0,1,0,0
71 GOSUB 180                                    'TAKEOFF
PERFORMANCE
81 GOSUB 1930                                   'CLIMB
PERFORMANCE
90 'GOSUB TO CRUISE PERFORMANCE WITH AUTOCALC OF DESCENT
GROSS WT (DGW)
95 GOSUB 3060                                  'CRUISE
PERFORMANCE
100 GOSUB 5120                                  'DESCENT
PERFORMANCE
103 DIM NAVDATA(9,9),COORD(9,3),WAYPTS(9,3):CLS:COLOR
15,1,4:LOCATE 1,1:PRINT "~W=MPOPTION/":SCREEN ,,3,0:INPUT
"",SELECTFLAG%
```

```
104 SCREEN ,,0,0
105 IF SELECTFLAG%=1 THEN SCR.SS%=2:INIT.SS%=-1:GOSUB 60000
106 BOB1=FRE(0):BOB2=FRE(""):IF SELECTFLAG%=1 THEN CHAIN
"BOTHALF",,ALL
107 IF SELECTFLAG%=2 THEN CHAIN "BOTHALF",,ALL
109 IF SELECTFLAG%=3 THEN GOTO 130              'SHOULD
EXIT TO DOS
125 'GOSUB 8000                                 'PRINT
COORD COMPUTATIONS
130 CLS:COLOR 15,0,0:CLS:SYSTEM
131 '
140 OPTALT1=EXP(11.0605-(5.53315E-08*(TOGW-800)*DC)-
(1.25963E-05*(TOGW-800))+(4.57508E-14*(TOGW-800)*(DC^3)))
150 SCEILING=((-.64*((TOGW/1000)-.8))+((-
.0413*DC)+60.2677))*1000
160 CLBDATA(7)=SCEILING
170 CLBDATA(8)=OPTALT1
180 '********** SUBROUTINE TO COMPUTE TAKEOFF PERFORMANCE
**********
183 ON ERROR GOTO 8000
185 COLOR 15,1,4
190 CLS
200 LOCATE 5,29,0
210 PRINT "TAKEOFF PERFORMANCE"
220 TOP = 3
230 LEFT = 27
240 WIDTZ = 23
250 HEIGHT = 5
260 GOSUB 7200                                  'box the
title
270 LOCATE 10,5,0
280 PRINT "TAKEOFF DATA WILL BE COMPUTED FOR THE FOLLOWING
CONDITIONS:"
290 PRINT
300 PRINT TAB(5) "TAKEOFF GROUND ROLL DISTANCE AND TAKEOFF
SPEED -----"
310 PRINT TAB(10) "* LEVEL, HARD SURFACE RUNWAY"
320 PRINT TAB(10) "* MILITARY RATED THRUST"
330 PRINT TAB(10) "* 1/2 FLAPS"
340 PRINT TAB(10) "* ZERO HEADWIND"
350 PRINT TAB(10) "* CG: 26% MAC"
360 PRINT
370 PRINT TAB(5) "MAXIMUM REFUSAL SPEED -----"
380 PRINT TAB(10) "* WITH ANTI-SKID"
385 IF OUTPUT=1 THEN GOSUB 7560
'AUTO PRINTSCREEN
390 LOCATE 24,27,0
400 PRINT "PRESS ANY KEY TO CONTINUE"
410 ANS$ = INKEY$: IF ANS$ = "" THEN 410
420 CLS
430 A = .2031573 + ((7.117837 * 10^-4) * PA)
```

```
440 B = (4.317391*10^-4)-((2.457174*10^-
5)*PA)+((6.404097*10^-9)*PA^2)-((4.946386*10^-13)*PA^3)
450 C = (5.22911*10^-4)+((2.528256*10^-7)*PA)-
((5.882059*10^-11)*PA^2)+((5.136734*10^-15)*PA^3)
460 GUIDE1 = A + (B*TEMP) + (C*TEMP^2)
470 TOFACTDD = ((GUIDE1 - 14)/(-1.75))+1.8
480 TOFACTNDD = ((GUIDE1 - 11.8)/(-1.478))+1.8
490 GRDD = (TOFACTDD-(13.13556-((1.564114*10^-
5)*TOGW)))/(6.22883*10^-3-((6.817641*10^-8)*TOGW)-
(166.1427/TOGW))
500 GRNDD = (TOFACTNDD-(13.13556-((1.56411*10^-
5)*TOGW)))/(6.22883*10^-3-((6.817641*10^-8)*TOGW)-
(166.1427/TOGW))
510 IF GRDD = 8000 THEN GRDD = GRDD - 100
520 IF (GRDD > 8000) AND (GRDD <= 9000) THEN GRDD = GRDD -
200
530 IF (GRDD > 9000) AND (GRDD <= 10000) THEN GRDD = GRDD -
300
540 IF (GRDD > 10000) AND (GRDD <= 11000) THEN GRDD = GRDD
- 400
550 IF (GRDD > 11000) AND (GRDD <= 12000) THEN GRDD = GRDD
- 500
560 IF (GRDD > 12000) AND (GRDD <= 13000) THEN GRDD = GRDD
- 600
570 IF GRDD > 13000 THEN GRDD = GRDD - 700
580 IF GRNDD = 8000 THEN GRNDD = GRNDD - 100
590 IF (GRNDD > 8000) AND (GRNDD <= 9000) THEN GRNDD =
GRNDD - 200
600 IF (GRNDD > 9000) AND (GRNDD <= 10000) THEN GRNDD =
GRNDD - 300
610 IF (GRNDD > 10000) AND (GRNDD <= 11000) THEN GRNDD =
GRNDD - 400
620 IF (GRNDD > 11000) AND (GRNDD <= 12000) THEN GRNDD =
GRNDD - 500
630 IF (GRNDD > 12000) AND (GRNDD <= 13000) THEN GRNDD =
GRNDD - 600
640 IF GRNDD > 13000 THEN GRNDD = GRNDD - 700
650 VTO = 75.15719+((2.21804*10^-3)*TOGW)+((1.967234*10^-
9)*TOGW^2)
660 D = (-11.45925)+((3.237367*10^-4)*TOGW)
670 E = (12.2164)-((1.72912*10^-4)*TOGW)
680 GUIDE2DD = EXP ((TOFACTDD - D)/ E)
690 GUIDE2NDD = EXP ((TOFACTNDD - D)/ E)
700 VREFDD = ((6.000001E-
04*RLENGTH+6.8704)*GUIDE2DD)+(.0031*RLENGTH)+59.6956
710 VREFNDD = ((6.000001E-
04*RLENGTH+6.8704)*GUIDE2NDD)+(.0031*RLENGTH)+59.6956
720 IF RLENGTH <= 12000 THEN VREFDD = VREFDD - 3
730 IF (RLENGTH > 12000) AND (RLENGTH <= 13000) THEN VREFDD
= VREFDD - 4
740 IF RLENGTH > 13000 THEN VREFDD = VREFDD - 6
```

```
750 IF RLENGTH = 12000 THEN VREFNDD = VREFNDD - 3
760 IF (RLENGTH > 12000) AND (RLENGTH <= 13000) THEN
VREFNDD = VREFNDD - 4
770 IF RLENGTH > 13000 THEN VREFNDD = VREFNDD - 6
780 TODATA(2,2) = GRDD
790 TODATA(2,3) = GRNDD
800 TODATA(3,2) = VTO
810 TODATA(3,3) = VTO
820 TODATA(4,2) = VREFDD
830 TODATA(4,3) = VREFNDD
840 '
850 'ECHO THE INPUT DATA AND PRINT THE COMPUTED TAKEOFF
PERFORMANCE
860 '
870 CLS
880 LOCATE 4,10: PRINT "INPUT DATA:"
890 LOCATE 6,10: PRINT "RUNWAY TEMPERATURE   =";:PRINT
USING "########";TEMP;
900 PRINT "  DEGREES FAHRENHEIT"
910 PRINT TAB(10) "PRESSURE ALTITUDE    =";:PRINT USING
"########";PA;
920 PRINT "  FEET"
930 PRINT TAB(10) "TAKEOFF GROSS WEIGHT = ";TOGW;" LBS."
940 PRINT TAB(10) "RUNWAY LENGTH        = ";RLENGTH;" FEET"
950 LOCATE 11,10: PRINT "COMPUTED TAKEOFF PERFORMANCE:"
960 LOCATE 13,31: PRINT "WITH DOUBLE DATUM" SPC(5) "WITHOUT
DOUBLE DATUM"
970 LOCATE 15,10: PRINT "TAKEOFF FACTOR"
980 PRINT TAB(10) "GROUND ROLL" SPC(16) "FEET" SPC(19)
"FEET"
990 PRINT TAB(10) "TAKEOFF SPEED" SPC(14) "KIAS" SPC(19)
"KIAS"
1000 PRINT TAB(10) "REFUSAL SPEED" SPC(14) "KIAS" SPC(19)
"KIAS"
1010 LOCATE 15,31: PRINT USING "##.#";TOFACTDD
1020 LOCATE 15,53: PRINT USING "##.#";TOFACTNDD
1030 LOCATE 16,31:PRINT USING "####";GRDD
1040 LOCATE 16,53:PRINT USING "####";GRNDD
1050 LOCATE 17,31:PRINT USING "####";VTO
1060 LOCATE 17,53:PRINT USING "####";VTO
1070 LOCATE 18,31:PRINT USING "####";VREFDD
1080 LOCATE 18,53:PRINT USING "####";VREFNDD
1090 TOP=10
1100 LEFT=7
1110 WIDTZ=68
1120 HEIGHT=9
1130 GOSUB 7200
1135 IF OUTPUT=1 THEN GOSUB 7560:LPRINT CHR$(12)
'AUTO PRINTSCREEN
1150 LOCATE 24,27,0:PRINT "PRESS ANY KEY TO CONTINUE"
1160 ANS$=INKEY$:IF ANS$= "" THEN 1160
```

```
1190 RETURN
1930 CLS
1940 DC = (DC7 + DC8)/2
1950 S = INT(403.56 - (.79075*DC) + (.0011382*DC^2) -
(4.1018E-07*DC^3))
1960 CLBDATA(1)=S
1970 IF DC > 150 THEN M = .86 - (.0021634*DC) + (7.6582E-
05*DC^2) - (1.1344E-06*DC^3) + (7.2125E-09*DC^4) -
(2.3035E-11*DC^5) + (3.6588E-14*DC^6) - (2.3062E-17*DC^7)
1980 IF DC > 150 THEN M = M*1000
1990 IF DC > 150 THEN M = INT(M)/1000 ELSE M =
ABS(.0052*(DC-150)) + (.0044*DC) + .04
2000 CLBDATA(2)=M
2010 CLBDATA(3)=HCRUISE
2020 '
2030 '***** COMPUTE THE CLIMB ... TIME *****
2040 '
2050 HCRUISE = HCRUISE/1000
2060 A = 406.9539*(HCRUISE^-.9542)
2070 IF HCRUISE < 30 THEN B = -.2384*HCRUISE + 29.6697
2080 IF (HCRUISE >= 30) AND (HCRUISE < 35) THEN B = -
.6089*HCRUISE + 40.9294
2090 IF HCRUISE >= 35 THEN B = -1.1572*HCRUISE + 60.1206
2100 INDEX = EXP(((TOGW/1000) - A)/B)
2110 IF HCRUISE < 5 THEN INDEX = .225
2120 IF (HCRUISE>5) AND (HCRUISE<7) THEN INDEX = INDEX +
(INDEX/2)
2130 IF (HCRUISE >= 7) AND (HCRUISE < 30) THEN INDEX =
INDEX - .15
2140 IF HCRUISE > 32 THEN INDEX = INDEX + .35
2150 IF DC <= 100 THEN TIME =
((.0155*DC)+2.4955)*(INDEX^((.0015*DC)+.9945))
2160 IF DC >= 200 THEN TIME =
((.0076*DC)+4.0455)*(INDEX^((.0015*DC)+.9945))
2170 IF (DC > 100) AND (DC <= 150) THEN TIME =
1.3532*EXP(((.004*DC)+.4495)*INDEX)
2180 IF (DC > 150) AND (DC < 200) THEN TIME = (-
.0027*DC+1.7806)*EXP(((.004*DC)+.4495)*INDEX)
2190 DEGC=ABS(TEMP-32)*.5556
2200 DELTAT=ABS(15-DEGC)/20
2210 '
2220 ' COMPUTE AND APPLY TEMPERATURE CORRECTIONS
2230 '
2240 IF TEMP < 59 THEN TIME = TIME - .5*DELTAT
2250 IF TEMP < 59 GOTO 2300
2260 IF TEMP = 59 THEN GOTO 2300
2270 IF TIME < 15 THEN TIME = TIME + 3*DELTAT
2280 IF (TIME >=15) AND (TIME < 20) THEN TIME = TIME +
2*DELTAT
2290 IF TIME >= 20 THEN TIME = TIME + DELTAT
2300 CLBDATA(4)=TIME
```

```
2310 '
2320 'PRINT THE RESULTS OF THE CLIMB COMPUTATIONS -- SPEED,
MACH, TIME
2330 '
2340 CLS
2350 LOCATE 6,30:PRINT "COMPUTED CLIMB DATA"
2360 LOCATE 10,20:PRINT "CLIMB SPEED    =  ";S;" KCAS TO
20,000 FEET"
2370 LOCATE 11,20:PRINT "CLIMB MACH     =  ";:PRINT USING
"#.##";M
2380 LOCATE 12,20:PRINT "FINAL ALTITUDE = ";:PRINT USING
"#####";HCRUISE*1000;
2390 PRINT "  FEET"
2400 LOCATE 13,20:PRINT "CLIMB TIME     =  ";:PRINT USING
"##.#";TIME;
2410 PRINT "  MINUTES"
2420 '
2430 '***** COMPUTE THE CLIMB...FUEL *****
2440 '
2450 HCRUISE=HCRUISE*1000
2460 IF HCRUISE<=5000 THEN INDEX2=(TOGW/1000)/28.5714
2470 IF (HCRUISE>5000) AND (HCRUISE<=10000) THEN
INDEX2=((TOGW/1000)-2.5)/12.5
2480 IF (HCRUISE>10000) AND (HCRUISE<=15000) THEN
INDEX2=((TOGW/1000)-1.8947)/8.421099
2490 IF (HCRUISE>15000) AND (HCRUISE<=20000) THEN
INDEX2=((TOGW/1000)-3.2047)/5.9347
2500 IF (HCRUISE>20000) AND (HCRUISE<=25000) THEN
INDEX2=((TOGW/1000)-3.6145)/4.8193
2510 IF (HCRUISE>5000) AND (HCRUISE<25000) THEN IF
(TOGW>40000!) AND (TOGW<=41000!) THEN INDEX2=INDEX2+.1 ELSE
IF TOGW>41000! THEN INDEX2=INDEX2+.15
2520 IF HCRUISE=25000 THEN IF TOGW>40000! THEN
INDEX2=INDEX2+.13
2530 IF HCRUISE>25000 THEN INDEX2=EXP(((TOGW/1000)-
((.6856*(HCRUISE/1000))-35.4857))/(-
.7754*(HCRUISE/1000)+48.0666))
2540 '
2550 'FUEL CALCULATION
2560 '
2570 IF DC<150 THEN B=.0011*DC+1.029 ELSE B=.0005*DC+1.1141
2580 IF DC<200 THEN A=.0015*DC+1.0163 ELSE
A=.0009*DC+1.1456
2590 FUEL=A*(INDEX2^B)
2600 '
2610 ' COMPUTE AND APPLY TEMPERATURE CORRECTIONS
2620 '
2630 IF TEMP=59 THEN FUEL=FUEL*100
2640 IF TEMP=59 GOTO 2720
2650 IF TEMP<59 GOTO 2710
2660 IF FUEL<3 THEN FUEL=(FUEL*100)+(55*DELTAT)
```

```
2670 IF (FUEL>=3) AND (FUEL<5) THEN
FUEL=(FUEL*100)+(90*DELTAT)
2680 IF (FUEL>=5) AND (FUEL<12) THEN
FUEL=(FUEL*100)+(120*DELTAT)
2690 IF (FUEL>=12) AND (FUEL<20) THEN
FUEL=(FUEL*100)+(140*DELTAT)
2700 IF TEMP > 59 GOTO 2720
2710 IF FUEL<=10 THEN FUEL=(FUEL-((FUEL/10)*DELTAT))*100
ELSE FUEL=(FUEL-(1.2*DELTAT))*100
2720 CLBDATA(5)=FUEL
2730 LOCATE 14,20:PRINT "FUEL REQUIRED  =  ";:PRINT USING
"####";FUEL;
2740 PRINT "  LBS"
2750 '
2760 '***** COMPUTE THE CLIMB ...DISTANCE *****
2770 '
2780 '"INDEX" IS USED FROM THE TIME CALCULATIONS CHARTS
2790 '
2800 INDEX=INDEX*20
2810 IF DC<200 THEN B=.0011*DC+1.0576 ELSE
B=.0003*DC+1.2085
2820 IF DC<200 THEN A=-.0009*DC+.9198 ELSE A=.7436
2830 IF (HCRUISE>25000) AND (HCRUISE<=30000) THEN A=A+.1
2840 IF HCRUISE>30000 THEN A=A+.2
2850 DIST=A*(INDEX^B)
2860 '
2870 ' COMPUTE AND APPLY TEMPERATURE CORRECTIONS
2880 '
2890 IF TEMP<59 GOTO 2920
2900 IF DIST<40 THEN DIST=DIST+8*DELTAT ELSE
DIST=DIST+12*DELTAT
2910 GOTO 2950
2920 IF DIST<=80 THEN DIST=DIST-((DIST/20)*DELTAT)
2930 IF DIST>140 THEN DIST=DIST-(5*DELTAT) ELSE DIST=DIST-
(8*DELTAT)
2940 CLBDATA(6)=DIST
2950 LOCATE 15,20:PRINT "DISTANCE       = ";:PRINT USING
"###.#";DIST;
2960 PRINT "  NM"
2970 TOP=3
2980 LEFT=15
2990 WIDTZ=50
3000 HEIGHT=15
3010 GOSUB 7200                          'box the output
3015 IF OUTPUT=1 THEN GOSUB 7560
3020 LOCATE 24,27,0:PRINT "PRESS ANY KEY TO CONTINUE"
3030 ANS$=INKEY$:IF ANS$= "" THEN 3030
3050 RETURN
3053 IF SELECTFLAG%=1 THEN GOSUB 3060
3054 IF SELECTFLAG%=1 THEN CHAIN "BOTHALF",8230,ALL
3060 '********** CRUISE PERFORMANCE SUBROUTINE **********
```

```
3200 CLS:LOCATE 10,20,0:PRINT "~W=COMPNOTE,NOWAIT/"
3210 '********** CRUISE OPTION 3 -- SPECIFIED ALTITUDE AND
MACH **********
3220 '
3230 I=1
3240 GOSUB 3710
3250 GOSUB 4960
3270 'CRUSDATA(1)=ALT
3280 'CRUSDATA(2)=CMACH                              'CRUISE
MACH NUMBER
3290 'CRUSDATA(3)=T                                  'AMBIENT
TEMP DEG RANKINE
3300 'CRUSDATA(4)=P                                  'AMBIENT
PRESSURE PSF
3310 'CRUSDATA(5)=SONIC                              'SPEED OF
SOUND a
3320 'CRUSDATA(6)=CL                                 'LIFT
COEFFICIENT
3330 'CRUSDATA(7)=CD                                 'DRAG
COEFFICIENT
3340 'CRUSDATA(8)=D                                  'TOTAL DRAG
LBS
3350 'CRUSDATA(9)=ENGTHRUST                          'ENGINE
THRUST LBS
3360 'CRUSDATA(10)=WF                                'FUEL FLOW
LBS/HR
3370 'CRUSDATA(11)=RBAR                              'SPECIFIC
RANGE NM/LB
3380 'CRUSDATA(12)=V                                 'GROUND
SPEED KTS
3390 'CRUSDATA(13)=H                                 'INDICATED
ALTITUDE
3400 'CRUSDATA(14)=MI                                'INDICATED
MACH NUMBER
3410 'CRUSDATA(15)=KIAS                              'INDICATED
AIRSPEED KTS
3420 'CRUSDATA(16)=DYNPRESS                          'DYNAMIC
PRESSURE q
3430 'CRUSDATA(17)=ALPHA                             'ANGLE OF
ATTACK IN RADIANS
3440 'CRUSDATA(18)=ALPHA0                            'AOA AT
CL=0
3450 'CRUSDATA(19)=ALPHA1
'1/(dALPHA/dCL)
3460 'CRUSDATA(20)=DELTATR                           'DEV FROM
STD DAY DEG RANKINE
3470 'CRUSDATA(21)=CDO                               'CDO
3480 'CRUSDATA(22)=CLF0                              'ZERO
THRUST CL
3490 'CRUSDATA(23)=CDF0                              'ZERO
THRUST CD
```

```
3500 'CRUSDATA(24)=DELCDS                          'STORE
TOTAL DRAG INCREMENT
3510 'CRUSDATA(25)=HDWIND                          'HEADWIND
KTS (NEGATIVE)
3520 '
3530 'PRINT RESULTS OF CRUISE CALCULATIONS
3540 '
3542 PRINT "~C=LAST/":CLS
3550 LOCATE 6,30:PRINT "COMPUTED CRUISE DATA"
3560 LOCATE 10,20:PRINT "SELECTED CRUISE ALTITUDE =
";ALT;:PRINT "FEET"
3570 LOCATE 11,20:PRINT "SELECTED CRUISE MACH     =
";:PRINT USING "#.##";CMACH
3580 LOCATE 12,20:PRINT "AMBIENT TEMPERATURE      =
";:PRINT USING "####";T-459.69;:PRINT " DEG F"
3590 LOCATE 13,20:PRINT "FUEL FLOW                =
";:PRINT USING "####";WF;:PRINT " LBS/HR"
3600 LOCATE 14,20:PRINT "SPECIFIC RANGE           =
";:PRINT USING "##.##";1/RBAR;:PRINT " LB/NM"
3610 LOCATE 15,20:PRINT "GROUND SPEED             =
";:PRINT USING "###";V;:PRINT " KTS"
3620 LOCATE 16,20:PRINT "TRUE AIRSPEED            =
";:PRINT USING "###";CMACH*SONIC*.5921;:PRINT " KTAS"
3630 TOP=3
3640 LEFT=15
3650 WIDTZ=50
3660 HEIGHT=15
3670 GOSUB 7200                                    'box the
output
3675 IF OUTPUT=1 THEN GOSUB 7560:LPRINT CHR$(12)
'AUTO PRINTSCREEN
3680 LOCATE 24,27,0:PRINT "PRESS ANY KEY TO CONTINUE"
3690 ANS$=INKEY$:IF ANS$= "" THEN 3690
3700 RETURN
3710 '********** SUBROUTINE PERF **********
3720 ALT=HCRUISE
3730 GOSUB 3800
3735 IF SELECTFLAG%=I THEN CMACH=LLKGS/(SONIC*.5921)
3740 GOSUB 3950
3750 ENGTHRUST=D/COS(AOA)
3760 GOSUB 4320
3770 V=.5921*CMACH*SONIC+HDWIND
3780 RBAR=V/(1.05*WF)
3790 RETURN
3800 '********** SUBROUTINE ATMOS **********
3810 '"ALT" IS THE SELECTED CRUISE ALTITUDE
3820 '"DELTATR IS THE TEMPERATURE DEVIATION IN DEGREES
RANKINE
3830 DELTATR=(TEMP+459.69)-518.69
3840 IF ALT>36089! GOTO 3900
3850 T=518.69-((3.5662*ALT)*10^-3)+DELTATR
```

```
3860 P=2116.2*(1-(6.8754E-06*ALT))^5.256
3870 SONIC=49.01*SQR(T)
3880 Q=262.5*P
3890 RETURN
3900 T=389.99+DELTATR
3910 P=472.68*EXP(-4.806E-05*(ALT-36089!))
3920 SONIC=49.01*SQR(T)
3930 Q=262.5*P
3940 RETURN
3950 '********** SUBROUTINE DRAG **********
3960 DEN=.4924-(.0239*CMACH)+(.3047*CMACH^2)
3970 ALPHA0=(.01935-(.02378*CMACH)+(.02756*CMACH^2))/DEN
3980 ALPHA1=.1396/DEN
3990 IF CMACH>.6 GOTO 4020
4000 DELCDS=DC6/10000
4010 GOTO 4140
4020 A1=DC6/10000+36*(DC7/10000-DC6/10000)
4030 B1=-120*(DC7/10000-DC6/10000)
4040 C1=100*(DC7/10000-DC6/10000)
4050 A2=DC7/10000+49*(DC8/10000-DC7/10000)-
5.6*(B1+(1.4*C1))
4060 B2=-140*(DC8/10000-DC7/10000)+15*(B1+(1.4*C1))
4070 C2=100*(DC8/10000-DC7/10000)-10*(B1+(1.4*C1))
4080 A3=DC8/10000+64*(DC9/10000-DC8/10000)-
7.2*(B2+(1.6*C2))
4090 B3=-160*(DC9/10000-DC8/10000)+17*(B2+(1.6*C2))
4100 C3=100*(DC9/10000-DC8/10000)-10*(B2+(1.6*C2))
4110 IF (CMACH>.6)AND (CMACH<.7) THEN
DELCDS=A1+(B1*CMACH)+(C1*CMACH^2)
4120 IF (CMACH>=.7) AND (CMACH<.8) THEN
DELCDS=A2+(B2*CMACH)+(C2*CMACH^2)
4130 IF CMACH>=.8 THEN DELCDS=A3+(B3*CMACH)+(C3*CMACH^2)
4140 IF CMACH>.825 THEN CDO=.015+.8889*((CMACH-.825)^2)
ELSE CDO=.015
4150 IF DESFLAG=1 THEN GOTO 4160 ELSE CLF0=(TOGW-
FUEL)/(Q*CMACH^2)
4160 CL=CLF0
4170 GOSUB 4270
4180 EF0=E
4190 CDF0=CDO+DELCDS+(CLF0^2)/(4*3.141593*EF0)
4200 CL=(CLF0-(CDF0*(ALPHA0-.0222)))/(1+(CDF0*ALPHA1))
4210 GOSUB 4270
4220 CD=CDO+DELCDS+(CL^2/(4*3.141593*E))
4230 ALPHA=ALPHA0+(CL*ALPHA1)
4240 AOA=ALPHA-.0222
4250 D=375*CD*(.7*CMACH^2*P)
4260 RETURN
4270 '********** SUBROUTINE E **********
4280 IF CL<=.3 THEN E=.875
4290 IF (CL>.3) AND (CL<=1.1) THEN E=.875-.783*((CL-
.3)^1.5)
```

```
4300 IF CL>1.1 THEN E=.3147
4310 RETURN
4320 '********** SUBROUTINE FUEL FLOW **********
4330 '
4340 'READ IN THRUST CALCULATION CONSTANTS INTO THRUST
ARRAY
4350 '
4352 RESTORE 4800
4360 FOR I=1 TO 9
4370  FOR J=1 TO 2
4380   READ THRUST(I,J)
4390  NEXT J
4400 NEXT I
4410 '
4420 'READ IN FUEL FLOW CALCULATION CONSTANTS INTO FUELFLO
ARRAY
4430 '
4431 RESTORE 4860
4440 FOR I=1 TO 18
4450  FOR J=1 TO 3
4460   READ FUELFLO(I,J)
4470  NEXT J
4480 NEXT I
4490 IF ALT<=20000 THEN J=1
'picks column altitude
4500 IF (ALT>20000) AND (ALT<=36089!) THEN J=2
4510 IF ALT>36089! THEN J=3
4520 K=1
4530 FOR I=1 TO 6
4540
W(I)=FUELFLO(K,J)+FUELFLO(K+1,J)*ALT+FUELFLO(K+2,J)*ALT^2
4550  K=K+3
4560 NEXT I
4570 W(7)=W(1)+W(2)*CMACH
4580 W(8)=W(3)+W(4)*CMACH
4590 W(9)=W(5)+W(6)*CMACH
4600 IF ALT<=36089! THEN J=1 ELSE J=2
'picks column thrust
4610 K=1
4620 FOR I=1 TO 3
4630
F(I)=THRUST(K,J)+THRUST(K+1,J)*ALT+THRUST(K+2,J)*ALT^2
4640  K=K+3
4650 NEXT I
4660 '
4670 'CALCULATIONS OF MAX THRUST AVAILABLE FOR SUBROUTINE
FCLIMB
4680 '
4690 'FPRIMEN=F(1)+F(2)*CMACH+F(3)*CMACH^2
4700 'TT=T*(1+(.2*CMACH^2))
```

```
4710 'IF TT<520 THEN ENGTHRUST=FPRIMEN ELSE
ENGTHRUST=FPRIMEN*(1-(.003875*(TT-520)*(1+(.2*CMACH^2))))
4720 '
4730 'FUEL FLOW CALCULATIONS FOR SUBROUTINE FUEL FLOW
4740 '
4750 WFSTD=W(7)+W(8)*ENGTHRUST+W(9)*ENGTHRUST^2
4760 WF=WFSTD*SQR(T/(T-DELTATR))
4770 '
4780 'THRUST COEFFICIENT DATA STATEMENTS
4790 '
4800 DATA 13712, 11852, -.4496, -.3358, 4.309E-6, 2.571E-6,
-6804, -6635
4810 DATA .2136, .1901, -2.004E-6, -1.493E-6, 9639, 9022, -
.2902, -.2456
4820 DATA 2.569E-6, 1.8226E-6
4830 '
4840 'FUEL FLOW COEFFICIENT DATA STATEMENTS
4850 '
4860 DATA 1.1481E3, 3.5298E3, -.6507E3, -2.0797E-2, -
2.3908E-1, 9.509E-2
4870 DATA -2.7714E-7, 4.6621E-6, -1.381E-6, 9.607E2, -
1.4681E3, 2.490E3
4880 DATA -8.3594E-2, 1.7707E-1, -1.378E-1, 2.8857E-6, -
4.0328E-6, 1.647E-6
4890 DATA 3.0843E-1, -2.4361, 4.679, -9.6006E-6, 2.2154E-4,
-2.319E-4
4900 DATA 1.3143E-10, -4.5717E-9, 2.521E-9, 4.0528E-1,
3.3845, -3.740
4910 DATA 2.1167E-5, -2.4055E-4, 2.105E-4, -8.3686E-10,
4.7787E-9, -2.243E-9
4920 DATA 2.0145E-5, 6.812E-4, -7.109E-4, 1.2972E-9, -
5.3767E-8, 2.647E-8
4930 DATA 4.4829E-14, 1.1523E-12, 0, 7.7595E-7, -7.1100E-4,
6.253E-4
4940 DATA -2.5781E-9, 5.7140E-8, -2.21E-8, 4.1052E-14, -
1.1675E-12,0
4950 RETURN
4960 '********** SUBROUTINE POS.ER. **********
4970 GOSUB 3800
4980 DYNPRESS=.7*P*CMACH^2
4990 IF CMACH>.65 THEN CPO=.4675-1.49*CMACH+(1.2*CMACH^2)
ELSE CPO=-.0616+.0231*CMACH+(.1244*CMACH^2)
5000 CPRIMEP=.000149+(.000511*CMACH)-(.00128*CMACH^2)
5010 PT=P*(1+.2*CMACH^2)^3.5
5020 CP=CPO+CPRIMEP*((TOGW-FUEL)/DYNPRESS)
5030 PII=P+CP*DYNPRESS
5040 GOSUB 5080
5050 MI=SQR(5*((PT/P)^.2857-1))
5060 KIAS=1478*SQR((1+(PT-PII)/2116.2)^.2857-1)
5070 RETURN
5080 '********** SUBROUTINE HFUNC (P) **********
```

```
5090 IF P<472.68 THEN H=36089!+20807!*(LOG(472.68/P)) ELSE
H=145450!*(1-((P/2116.2)^.1903))
5100 RETURN
5110 '
5120 '********** DESCENT MODE SUBROUTINE **********
5130 '
5140 DELR=0
5150 DELRJ=0
5160 DELW=0
5170 DELWJ=0
5180 CLS:LOCATE 10,20,0:PRINT "~W=COMPNOTE,NOWAIT/"
5190 DGW=TOGW-FUEL  'Modified from original program
5200 DKIAS=322-((49.5-.0007325*DGW)*(1/SQR(.015)-
(1/SQR(.015+DC6/10000))))
5210 MCL=.85
5220 GOSUB 5900                                    'SUBROUTINE
HM
5230 GOSUB 5960                                    'SUBROUTINE
DESC/ALTS
5240 K=0
5250 G=0
5260 K=K+1
5270 H=DESCH(K)
5280 ALT=H
5290 GOSUB 3800                                    'SUBROUTINE
ATMOS
5300 IF H>=HM THEN M=.85 ELSE
M=2.236*SQR((1+(2116.2/P)*((1+4.571E-07*DKIAS^2)^3.5-
1))^.2857-1)
5310 IF G=1 THEN GOTO 5620                         'SUBROUTINE C
5320 GOSUB 6290                                    'SUBROUTINE
DESC/DER
5330 H=H+1
5340 GOSUB 6240                                    'SUBROUTINE U
5350 IF DESCH(K)<>HSD GOTO 5410
5360 HJ=DESCH(K)
5370 RPJ=RP
5380 WPJ=WP
5390 UJ=U
5400 GOTO 5260
5410 HJ1=DESCH(K)
5420 RPJ1=RP
5430 WPJ1=WP
5440 UJ1=U
5450 DELRJ=8.229E-05*(UJ*RPJ+UJ1*RPJ1)*(HJ1-HJ)
5460 DELWJ=.0001458*(UJ*RPJ*WPJ+UJ1*RPJ1*WPJ1)*(HJ1-HJ)
5470 IF DESCH(K)<>HED THEN GOTO 5530
5480 HBARJ=.5*(HJ+HJ1)
5490 H=HBARJ
5500 ALT=DESCH(K)
5510 G=1
```

```
5520 GOTO 5290
5530 H=H-2
5540 GOSUB 6240                                    'SUBROUTINE U
5550 HJ=DESCH(K)
5560 RPJ=RPJ1
5570 WPJ=WPJ1
5580 UJ=U
5590 DELR=DELR+DELRJ
5600 DELW=DELW+DELWJ
5610 GOTO 5260
5620 GOSUB 6530                                    'SUBROUTINE
WFIDL
5630 WBARPJ=WFIDL/(M*SONIC+(HDWIND/.5921))
5640 IF HJ1=HED THEN NJ=1/3+(4/3)*(WBARPJ/(WPJ+WPJ1)) ELSE
NJ=1
5650 DELWJ=DELWJ*NJ
5660 DELR=DELR+DELRJ
5670 DELW=DELW+DELWJ
5680 DESCTIME=HSD/3200
5690 DESDATA(1)=DELW
5700 DESDATA(2)=DELR
5710 DESDATA(3)=DKIAS
5720 DESDATA(4)=DESCTIME
5730 'PRINT RESULTS
5740 '
5750 PRINT "~C=LAST/":CLS
5760 LOCATE 5,30:PRINT "COMPUTED DESCENT DATA"
5770 LOCATE 8,27:PRINT "DESCENT FUEL USED = ";:PRINT USING
"#####";DELW;:PRINT " LBS"
5780 LOCATE 10,27:PRINT "DISTANCE COVERED  = ";:PRINT USING
"##.#";DELR;:PRINT " NM"
5790 LOCATE 12,27:PRINT "DESCENT TIME      = ";:PRINT USING
"##.#";DESCTIME;:PRINT " MIN"
5800 LOCATE 14,27:PRINT "DESCENT SPEED     =  ";:PRINT
USING "###";DKIAS;:PRINT " KIAS"
5810 TOP=3
5820 LEFT=15
5830 WITDZ=40
5840 HEIGHT=13
5850 GOSUB 7200                                        'box
the output
5855 IF OUTPUT=1 THEN GOSUB 7560:LPRINT CHR$(12)
'AUTO PRINTSCREEN
5860 LOCATE 24,27,0:PRINT "PRESS ANY KEY TO CONTINUE"
5870 ANS$=INKEY$:IF ANS$= "" THEN 5870
5875 ERASE CLBDATA,DESCH,DESDATA
5880 RETURN
5890 STOP
5900 '********** SUBROUTINE HM **********
5910 P=2116.2*((1+(4.571E-07*DKIAS^2))^3.5-
1)/((1+.2*MCL^2)^3.5-1)        'PM
```

```
5920 GOSUB 5080                                    'SUBROUTINE
HFUNC(P)
5930 HM=H
5940 RETURN
5950 '
5960 '********** SUBROUTINE DESC/ALTS **********
5970 '
5980 IF (HM>HED) AND (HM<HSD) THEN GOTO 6070
5990 IF (36089!>HED) AND (36089!<HSD) THEN GOTO 6040
6000 DESCH(1)=HSD
6010 DESCH(2)=36089!
6020 DESCH(3)=HED
6030 RETURN
6040 DESCH(1)=HSD
6050 DESCH(2)=HED
6060 RETURN
6070 IF (36089!>HED) AND (36089!<HSD) THEN GOTO 6120
6080 DESCH(1)=HSD
6090 DESCH(2)=HM
6100 DESCH(3)=HED
6110 RETURN
6120 IF HM>36089! THEN GOTO 6180
6130 DESCH(1)=HSD
6140 DESCH(2)=36089!
6150 DESCH(3)=HM
6160 DESCH(4)=HED
6170 RETURN
6180 DESCH(1)=HSD
6190 DESCH(2)=HM
6200 DESCH(3)=36089!
6210 DESCH(4)=HED
6220 RETURN
6230 '
6240 '********** SUBROUTINE U **********
6250 IF H>=HM THEN U=1 ELSE U=2+.2*M^2-(1+.2*M^2)^-2.5
6260 IF H>=36089! THEN RETURN ELSE U=U-.133*M^2
6270 RETURN
6280 '
6290 '********** SUBROUTINE DESC/DER **********
6300 CMACH=M
6310 CLF0=DGW/(Q*M^2)
6320 DESFLAG=1
6330 GOSUB 3950
6340 DESFLAG=0
6350 GOSUB 6420
6360 GOSUB 6530
6370 V=M*SONIC
6380 RP=(DGW/(FIDL-D))*(1+((HDWIND/.5921)/V))*U
6390 WP=WFIDL/((HDWIND/.5921)+V)
6400 RETURN
6410 '
```

```
6420 '********** SUBROUTINE FIDL **********
6430 IF H>36089! THEN GOTO 6480
6440 FO=668+.01058*H
6450 FT=-2.4-.0000277*H
6460 FIDL=FO-950*M+FT*DELTATR
6470 RETURN
6480 FO=-78+.03127*H
6490 FT=13.7-.000473*H
6500 FIDL=FO-950*M+FT*DELTATR
6510 RETURN
6520 '
6530 '********** SUBROUTINE WFIDL **********
6540 IF H>36089! THEN GOTO 6600
6550 WFO=928-.0183*H+3.38E-07*H^2
6560 WFM=1150-.0319*H
6570 WFT=-1.25
6580 WFIDL=WFO+WFM*M+WFT*DELTATR
6590 RETURN
6600 WFO=492+.00597*H
6610 WFM=0
6620 WFT=8.5-.00027*H
6630 WFIDL=WFO+WFM*M+WFT*DELTATR
6640 RETURN
6650 '
6652 END
7200 '********** SUBROUTINE DBLBOX **********
7210 ' TOP = ROW NUMBER OF UPPER LEFT CORNER
7220 ' LEFT = COLUMN OF UPPER LEFT CORNER
7230 ' WIDTZ = WIDTH OF BOX
7240 ' HEIGHT = HEIGHT OF BOX
7250 '
7260 BOTTOM = TOP + HEIGHT
7270 RIGHT = LEFT + WIDTZ
7280 '
7290 ' LOCATE AND PRINT THE CORNERS OF THE BOX
7300 '
7310 LOCATE  TOP, LEFT
7320 PRINT  CHR$(201)          'PRINTs upper left corner
7330 LOCATE  TOP, RIGHT
7340 PRINT CHR$(187)           'PRINTs upper right corner
7350 LOCATE BOTTOM, LEFT
7360 PRINT CHR$(200)           'PRINTs lower left corner
7370 LOCATE BOTTOM, RIGHT
7380 PRINT CHR$(188)           'PRINTs lower right corner
7390 '
7400 ' DRAW THE TOP, BOTTOM, AND SIDES OF THE BOX
7410 '
7420 FOR COL = (LEFT + 1) TO (RIGHT - 1)       'top and
bottom
7430  LOCATE TOP, COL
7440  PRINT CHR$(205)
```

```
7450  LOCATE BOTTOM, COL
7460  PRINT CHR$(205)
7470 NEXT COL
7480 FOR ROW = (TOP + 1) TO (BOTTOM - 1)       'left and
right sides
7490  LOCATE ROW, LEFT
7500  PRINT CHR$(186)
7510  LOCATE ROW, RIGHT
7520  PRINT CHR$(186)
7530 NEXT ROW
7540 RETURN
7550 '
7560 '********** PRINT SCREEN ASSEMBLY LANGUAGE SUBROUTINE
**********
7580 A%(0)=&HCD55
7590 A%(1)=&H5D05
7600 A%(2)=&H90CB
7610 SUBRT = VARPTR(A%(0))
7620 CALL SUBRT
7640 RETURN
7700 '********** INTRODUCTION AND SETUP SUBROUTINE
**********
7710 LOCATE 1,1,0:PRINT "~W=INTROSEL/":SCREEN ,,3,0:INPUT
"",INTRO:SCREEN ,,0,0
7720 IF INTRO=3 THEN RETURN
7730 IF INTRO=2 THEN PRINT "~W=INTRO5/":SCREEN ,,3,0:INPUT
"",OUTPUT:SCREEN ,,0,0
7745 IF INTRO=1 THEN PRINT "~W=INTRO/":SCREEN ,,3,0:INPUT
"",OUTPUT:SCREEN ,,0,0
7750 RETURN
8000 '********** ERROR TRAP FOR PRINTER ERRORS **********
8010 IF ERR=27 THEN PRINT "~W=PRINTERR/"
8020 ON ERROR GOTO 0:RESUME
10000 ON ERROR GOTO 62980     ' Error Handling Routine -
Delete
10010                         ' if it conflicts with your
own
30000 '
30005 '    Variables Section For B:XCINPUT
30010 '
30015 VL.SS$(1)=STR$(TEMP): VL.SS$(2)=STR$(PA):
VL.SS$(3)=STR$(TOGW):
30020 VL.SS$(4)=STR$(RLENGTH): VL.SS$(5)=STR$(DC6):
VL.SS$(6)=STR$(DC7):
30025 VL.SS$(7)=STR$(DC8): VL.SS$(8)=STR$(DC9):
VL.SS$(9)=STR$(CMACH):
30030 VL.SS$(10)=STR$(HCRUISE): VL.SS$(11)=STR$(HDWIND):
VL.SS$(12)=STR$(HSD):
30035 VL.SS$(13)=STR$(HED): VL.SS$(14)=STR$(FUELLOAD):
30040 RETURN
30045 '
```

```
30050 '    Assign VL.SS$ Array to the variables
30055 '
30060
TEMP=VAL(VL.SS$(1)):PA=VAL(VL.SS$(2)):TOGW=VAL(VL.SS$(3)):
30065
RLENGTH=VAL(VL.SS$(4)):DC6=VAL(VL.SS$(5)):DC7=VAL(VL.SS$(6)
):
30070
DC8=VAL(VL.SS$(7)):DC9=VAL(VL.SS$(8)):CMACH=VAL(VL.SS$(9)):
30075
HCRUISE=VAL(VL.SS$(10)):HDWIND=VAL(VL.SS$(11)):HSD=VAL(VL.S
S$(12)):
30080 HED=VAL(VL.SS$(13)):FUELLOAD=VAL(VL.SS$(14)):
30085 RETURN
30090 '
30095 '     Section To Initialize Variables To Initial
Values
30100 '
30105 TEMP=0: PA=0: TOGW=31200: RLENGTH=13500: DC6=0:
30110 DC7=0: DC8=0: DC9=0: CMACH=0!: HCRUISE=27000:
30115 HDWIND=0: HSD=27000: HED=0: FUELLOAD=10000:
30120 RETURN
30125 '
30130 '     ****  List DATA statements & Print DISPLAY Only
Variables  ****
30135 'Lin,Col,Len,Picture,Low Range,High
Range,Foreground,Background,# of Edit
30140 '
30145 DATA 53,5,6,"N","######",0,120,15,0,0
30150 DATA 53,6,6,"N","######",0,8000,15,0,0
30155 DATA 53,7,6,"N","######",25000,42000,15,0,0
30160 DATA 53,8,6,"N","######",0,99999,15,0,0
30165 DATA 53,9,6,"N","###XXX",0,999,15,0,0
30170 DATA 53,10,6,"N","######",0,999,15,0,0
30175 DATA 53,11,6,"N","######",0,999,15,0,0
30180 DATA 53,12,6,"N","######",0,999,15,0,0
30185 DATA 53,13,6,"N","###.##",0.00,1.20,15,0,1
30190 DATA 53,14,6,"N","######",100,46000,15,0,0
30195 DATA 53,15,6,"N","######",-200,200,15,0,0
30200 DATA 53,16,6,"N","######",100,46000,15,0,0
30205 DATA 53,17,6,"N","######",0,46000,15,0,0
30210 DATA 53,19,6,"N","######",0,20000,15,0,0
30215 RETURN
30220 '
30225 '    Screen Display Initialization Statements
30230 '
30235 NUMFLDS.SS%=14: FILNM.SS$="XCINPUT.SCR":
30240 EXITCHR.SS$=CHR$(27)+CHR$(127)+""
30245 RESTORE 30130
30250 RETURN
30255 '
```

```
30260 '    Variables Section For B:LLINPUT
30265 '
30270 VL.SS$(1)=STR$(LLALT): VL.SS$(2)=STR$(LLKGS):
VL.SS$(3)=STR$(LLFUEL1):
30275 RETURN
30280 '
30285 '    Assign VL.SS$ Array to the variables
30290 '
30295
LLALT=VAL(VL.SS$(1)):LLKGS=VAL(VL.SS$(2)):LLFUEL1=VAL(VL.SS
$(3)):
30300 RETURN
30305 '
30310 '     Section To Initialize Variables To Initial
Values
30315 '
30320 LLALT=200: LLKGS=360: LLFUEL1=0:
30325 RETURN
30330 '
30335 '     ****  List DATA statements & Print DISPLAY Only
Variables  ****
30340 'Lin,Col,Len,Picture,Low Range,High
Range,Foreground,Background,# of Edit
30345 '
30350 DATA 48,8,5,"N","#####",100,45000,15,0,0
30355 DATA 48,11,5,"N","#####",0,700,15,0,0
30360 DATA 48,14,5,"N","#####",0,16000,15,0,0
30365 RETURN
30370 '
30375 '    Screen Display Initialization Statements
30380 '
30385 NUMFLDS.SS%=3: FILNM.SS$="LLINPUT.SCR":
30390 EXITCHR.SS$=CHR$(27)+CHR$(127)+""
30395 RESTORE 30335
30400 RETURN
30405 '
30410 '    Variables Section For B:LLJETLOG
30415 '
30420 RETURN
30425 '
30430 '    Assign VL.SS$ Array to the variables
30435 '
30440 RETURN
30445 '
30450 '     Section To Initialize Variables To Initial
Values
30455 '
30460 RETURN
30465 '
30470 '     ****  List DATA statements & Print DISPLAY Only
Variables  ****
```

```
30475 'Lin,Col,Len,Picture,Low Range,High
Range,Foreground,Background,# of Edit
30480 '
30485 COLOR 15,0: LOCATE 4,13: PRINT
LEFT$(GEOLAT$+LEFT$(BLNK.SS$,1-LEN(GEOLAT$)),1);
30490 COLOR 15,0: LOCATE 4,17: PRINT
LEFT$(GEOLONG$+LEFT$(BLNK.SS$,1-LEN(GEOLONG$)),1);
30495 COLOR 15,0: LOCATE 4,32: PRINT
LEFT$(MVARTYPE$+LEFT$(BLNK.SS$,1-LEN(MVARTYPE$)),1);
30500 COLOR 14,1: LOCATE 6,12: PRINT USING "####.####"; A1;
30505 COLOR 14,1: LOCATE 7,12: PRINT USING "####.####"; A2;
30510 COLOR 14,1: LOCATE 8,12: PRINT USING "####.####"; A3;
30515 COLOR 14,1: LOCATE 9,12: PRINT USING "####.####"; A4;
30520 COLOR 14,1: LOCATE 10,12: PRINT USING "####.####";
A5;
30525 COLOR 14,1: LOCATE 11,12: PRINT USING "####.####";
A6;
30530 COLOR 14,1: LOCATE 12,12: PRINT USING "####.####";
A7;
30535 COLOR 14,1: LOCATE 13,12: PRINT USING "####.####";
A8;
30540 COLOR 14,1: LOCATE 14,12: PRINT USING "####.####";
A9;
30545 COLOR 14,1: LOCATE 15,12: PRINT USING "####.####";
A10;
30550 COLOR 14,1: LOCATE 16,12: PRINT USING "####.####";
A11;
30555 COLOR 14,1: LOCATE 17,12: PRINT USING "####.####";
A12;
30560 COLOR 14,1: LOCATE 18,12: PRINT USING "####.####";
A13;
30565 COLOR 14,1: LOCATE 19,12: PRINT USING "####.####";
A14;
30570 COLOR 14,1: LOCATE 20,12: PRINT USING "####.####";
A15;
30575 COLOR 14,1: LOCATE 21,12: PRINT USING "####.####";
A16;
30580 COLOR 14,1: LOCATE 22,12: PRINT USING "####.####";
A17;
30585 COLOR 14,1: LOCATE 23,12: PRINT USING "####.####";
A18;
30590 COLOR 14,1: LOCATE 6,27: PRINT USING "######"; B1;
30595 COLOR 14,1: LOCATE 8,27: PRINT USING "######"; B2;
30600 COLOR 14,1: LOCATE 10,27: PRINT USING "######"; B3;
30605 COLOR 14,1: LOCATE 12,27: PRINT USING "######"; B4;
30610 COLOR 14,1: LOCATE 14,27: PRINT USING "######"; B5;
30615 COLOR 14,1: LOCATE 16,27: PRINT USING "######"; B6;
30620 COLOR 14,1: LOCATE 18,27: PRINT USING "######"; B7;
30625 COLOR 14,1: LOCATE 20,27: PRINT USING "######"; B8;
30630 COLOR 14,1: LOCATE 22,27: PRINT USING "######"; B9;
30635 COLOR 14,1: LOCATE 8,36: PRINT USING "########"; C1;
```

```
30640 COLOR 14,1: LOCATE 9,36: PRINT USING "######.#"; C2;
30645 COLOR 14,1: LOCATE 10,36: PRINT USING "########"; C3;
30650 COLOR 14,1: LOCATE 11,36: PRINT USING "######.#"; C4;
30655 COLOR 14,1: LOCATE 12,36: PRINT USING "########"; C5;
30660 COLOR 14,1: LOCATE 13,36: PRINT USING "######.#"; C6;
30665 COLOR 14,1: LOCATE 14,36: PRINT USING "########"; C7;
30670 COLOR 14,1: LOCATE 15,36: PRINT USING "######.#"; C8;
30675 COLOR 14,1: LOCATE 16,36: PRINT USING "########"; C9;
30680 COLOR 14,1: LOCATE 17,36: PRINT USING "######.#";
C10;
30685 COLOR 14,1: LOCATE 18,36: PRINT USING "########";
C11;
30690 COLOR 14,1: LOCATE 19,36: PRINT USING "######.#";
C12;
30695 COLOR 14,1: LOCATE 20,36: PRINT USING "########";
C13;
30700 COLOR 14,1: LOCATE 21,36: PRINT USING "######.#";
C14;
30705 COLOR 14,1: LOCATE 22,36: PRINT USING "########";
C15;
30710 COLOR 14,1: LOCATE 23,36: PRINT USING "######.#";
C16;
30715 COLOR 14,1: LOCATE 8,47: PRINT USING "#####"; D1;
30720 COLOR 14,1: LOCATE 8,53: PRINT USING "##"; D2;
30725 COLOR 14,1: LOCATE 9,47: PRINT USING "#####"; D3;
30730 COLOR 14,1: LOCATE 9,53: PRINT USING "##"; D4;
30735 COLOR 14,1: LOCATE 10,47: PRINT USING "#####"; D5;
30740 COLOR 14,1: LOCATE 10,53: PRINT USING "##"; D6;
30745 COLOR 14,1: LOCATE 11,47: PRINT USING "#####"; D7;
30750 COLOR 14,1: LOCATE 11,53: PRINT USING "##"; D8;
30755 COLOR 14,1: LOCATE 12,47: PRINT USING "#####"; D9;
30760 COLOR 14,1: LOCATE 12,53: PRINT USING "##"; D10;
30765 COLOR 14,1: LOCATE 13,47: PRINT USING "#####"; D11;
30770 COLOR 14,1: LOCATE 13,53: PRINT USING "##"; D12;
30775 COLOR 14,1: LOCATE 14,47: PRINT USING "#####"; D13;
30780 COLOR 14,1: LOCATE 14,53: PRINT USING "##"; D14;
30785 COLOR 14,1: LOCATE 15,47: PRINT USING "#####"; D15;
30790 COLOR 14,1: LOCATE 15,53: PRINT USING "##"; D16;
30795 COLOR 14,1: LOCATE 16,47: PRINT USING "#####"; D17;
30800 COLOR 14,1: LOCATE 16,53: PRINT USING "##"; D18;
30805 COLOR 14,1: LOCATE 17,47: PRINT USING "#####"; D19;
30810 COLOR 14,1: LOCATE 17,53: PRINT USING "##"; D20;
30815 COLOR 14,1: LOCATE 18,47: PRINT USING "#####"; D21;
30820 COLOR 14,1: LOCATE 18,53: PRINT USING "##"; D22;
30825 COLOR 14,1: LOCATE 19,47: PRINT USING "#####"; D23;
30830 COLOR 14,1: LOCATE 19,53: PRINT USING "##"; D24;
30835 COLOR 14,1: LOCATE 20,47: PRINT USING "#####"; D25;
30840 COLOR 14,1: LOCATE 20,53: PRINT USING "##"; D26;
30845 COLOR 14,1: LOCATE 21,47: PRINT USING "#####"; D27;
30850 COLOR 14,1: LOCATE 21,53: PRINT USING "##"; D28;
30855 COLOR 14,1: LOCATE 22,47: PRINT USING "#####"; D29;
```

```
30860 COLOR 14,1: LOCATE 22,53: PRINT USING "##"; D30;
30865 COLOR 14,1: LOCATE 23,47: PRINT USING "#####"; D31;
30870 COLOR 14,1: LOCATE 23,53: PRINT USING "##"; D32;
30875 COLOR 14,1: LOCATE 6,69: PRINT USING "#########"; E1;
30880 COLOR 14,1: LOCATE 8,58: PRINT USING "########"; E2;
30885 COLOR 14,1: LOCATE 8,69: PRINT USING "#########"; E3;
30890 COLOR 14,1: LOCATE 10,58: PRINT USING "########"; E4;
30895 COLOR 14,1: LOCATE 10,69: PRINT USING "#########";
E5;
30900 COLOR 14,1: LOCATE 12,58: PRINT USING "########"; E6;
30905 COLOR 14,1: LOCATE 12,69: PRINT USING "#########";
E7;
30910 COLOR 14,1: LOCATE 14,58: PRINT USING "########"; E8;
30915 COLOR 14,1: LOCATE 14,69: PRINT USING "#########";
E9;
30920 COLOR 14,1: LOCATE 16,58: PRINT USING "########";
E10;
30925 COLOR 14,1: LOCATE 16,69: PRINT USING "#########";
E11;
30930 COLOR 14,1: LOCATE 18,58: PRINT USING "########";
E12;
30935 COLOR 14,1: LOCATE 18,69: PRINT USING "#########";
E13;
30940 COLOR 14,1: LOCATE 20,58: PRINT USING "########";
E14;
30945 COLOR 14,1: LOCATE 20,69: PRINT USING "#########";
E15;
30950 COLOR 14,1: LOCATE 22,58: PRINT USING "########";
E16;
30955 COLOR 14,1: LOCATE 22,69: PRINT USING "#########";
E17;
30960 COLOR 15,1: LOCATE 24,36: PRINT USING "######.#";
TOTALDIST;
30965 COLOR 15,1: LOCATE 24,47: PRINT USING "#####"; F1;
30970 COLOR 15,1: LOCATE 24,53: PRINT USING "##"; F2;
30975 COLOR 15,1: LOCATE 24,58: PRINT USING "########"; F3;
30980 COLOR 15,1: LOCATE 24,69: PRINT USING "#########";
F4;
30985 RETURN
30990 '
30995 '    Screen Display Initialization Statements
31000 '
31005 NUMFLDS.SS%=0: FILNM.SS$="LLJETLOG.SCR":
31010 EXITCHR.SS$=CHR$(27)+CHR$(127)+""
31015 RESTORE 30470
31020 RETURN
60000 '
60010 '
èëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëë
£
```

```
60020 ' ¤   START OF SCREEN SCULPTOR FULL SCREEN EDITING
ROUTINE   ¤
60030 '
àëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëë
¥
60050 '               Ö----------------------------¢
60060 '               °   Main body of subroutine   °
60070 '               â----------------------------ì
60080 '
60085 IF SAME.SS% THEN 60200  'To return to the SAME screen
with SAME values
60090 IF SCR.SS%=SCRLST.SS% THEN 60140  ' If same screen as
last, don't reload
60100                                  ' Get screen setup
parameters
60105    ON SCR.SS% GOSUB   30225, 30375, 30995
60110    GOSUB 60550                   ' Read field data
for this screen
60120    OUT &H3D8,&H1                 ' Turn off screen
display
60125 '  CALL QUBLOAD(FILNM.SS$)       '<- For QuickBASIC,
See READ.ME file
60130    DEF SEG=SCRNSEG.SS% : BLOAD FILNM.SS$,0 :DEF SEG '
Load screen picture
60135                   ' Set initial values
60140    IF INIT.SS% THEN ON SCR.SS% GOSUB   30095, 30310,
30450
60145                   ' Assign current values to screen
array
60155    ON SCR.SS% GOSUB   30005, 30260, 30410
60160    OUT &H3D8,&H29                ' Turn on screen
display
60170    GOSUB 60590                   ' Pad fields with
blanks and display
60175                   ' Display initial DISPLAY variables
60180    ON SCR.SS% GOSUB   30130, 30335, 30470
60185    OUT &H3D8,&H29                ' Turn on screen
display
60190 '
60195 F.SS%=1 : SCRLST.SS%=SCR.SS%
60200 COLOR 7,0:LOCATE 25,1:PRINT BLNK.SS$;   'Clr msg from
prior screen
60205 IF NUMFLDS.SS%=0 THEN RETURN            'Exit if no
fields on screen
60210 LOCATE ,,,0,13                          'Make cursor
size large
60215 EXSCR.SS%=0                             'Initialize
Flag For Screen Exit
60220 WHILE NOT EXSCR.SS%        'Loop on each field(until
Exit Flag is set)
```

```
60250     GOSUB 60740                          'Accept input
data for this field
60260     '
60265     ' The above subroutine accepts data for a single
field.  The                        program will return to
this spot after the cursor exits any
field on the input screen.
60270     '
60275     ' If you want to do any special input field
testing, this is
60280     ' a good place to do it.
60285     '
60290     ' The following variables are passed back for
your use:
60295     ' F.SS% is next field to be edited
60300     ' FLDLST.SS% is last field edited
60305     ' LASTCHR.SS$ is last keyboard character entered
60310     ' VL.SS$(n) contains the current value of field
n.
60315     '
60320 WEND
60360 '
60370 '
60390 COLOR 7,0:LOCATE 25,1:PRINT BLNK.SS$;:LOCATE
25,15:PRINT "... Please WAIT A Moment While Checking Fields
...";
60400 FOR F.SS%=1 TO NUMFLDS.SS%     ' Test Each Field
Before Leaving Screen
60410     GOSUB 62500                ' Check contents of
this field
60420     IF ERR.MSG%=-1 THEN EXSCR.SS%=0 : GOTO 60220
'Error Detected
60430 NEXT F.SS%
60440 F.SS%=FLDLST.SS%                         'Reset Field
indicator
60450 '
60455                                          'Assign new
field values
60460 ON SCR.SS% GOSUB   30050, 30285, 30430
60470 RETURN                                   'Exit this
subroutine
60480 '
60490
'************************************************************
**************
60500 '
60510 '
60520 ' ****************************************
60530 ' **** Read Field Data For This Screen ****
60540 ' ****************************************
60550 FOR F.SS%=1 TO NUMFLDS.SS%
```

```
60555   READ
LO.SS%(F.SS%,2),LO.SS%(F.SS%,1),LE.SS%(F.SS%),TY.SS$(F.SS%)
,PIC.SS$(F.SS%),RG.SS(F.SS%,1),RG.SS(F.SS%,2),CL.SS%(F.SS%,
1),CL.SS%(F.SS%,2),SPECCHR.SS%(F.SS%)
60560 NEXT F.SS%
60565 RETURN
60570 '
60575
'**************************************************************
**************
60580 '*** Pad Fields With Blanks, Insert Special
Characters, and Display Flds
60585
'**************************************************************
**************
60590 FOR F.SS%=1 TO NUMFLDS.SS%
60595     IF TY.SS$(F.SS%)="N" THEN 60655   ' This section
for non-numeric types
60600        IF LEN(VL.SS$(F.SS%))>LE.SS%(F.SS%) THEN
VL.SS$(F.SS%)=LEFT$(VL.SS$(F.SS%),LE.SS%(F.SS%)): GOTO
60610
60605
VL.SS$(F.SS%)=VL.SS$(F.SS%)+MID$(BLNK.SS$,1,LE.SS%(F.SS%)-
LEN(VL.SS$(F.SS%)))
60610        IF INSTR("CD",TY.SS$(F.SS%))=0 OR
SPECCHR.SS%(F.SS%)=0 THEN 60690
60615           CNT.SS%=0
60620           FOR J.SS%=1 TO LE.SS%(F.SS%)           '
Insert Special chars
60625              IF
INSTR("ULX#89",MID$(PIC.SS$(F.SS%),J.SS%,1))=0 THEN
MID$(VL.SS$(F.SS%),J.SS%,1)=MID$(PIC.SS$(F.SS%),J.SS%,1) :
CNT.SS%=CNT.SS%+1
60630              IF CNT.SS%=SPECCHR.SS%(F.SS%) THEN
60690
60635           NEXT J.SS%
60640        GOTO 60690          ' End of "non-numeric
type" section
60645 '
60650 '                          ' The following section is
for numeric types
60655     NUMDEC%=LE.SS%(F.SS%)-INSTR(PIC.SS$(F.SS%),".")
' Calc # of dec places
60657     IF LEFT$(VL.SS$(F.SS%),1)=" " THEN
VL.SS$(F.SS%)=RIGHT$(VL.SS$(F.SS%),LEN(VL.SS$(F.SS%))-1) '
Strip leading blank
60660     IF NUMDEC%=LE.SS%(F.SS%) THEN NUMDEC%=0:
NUMINT%=LE.SS%(F.SS%) ELSE NUMINT%=LE.SS%(F.SS%)-NUMDEC%-1
' Calc # of interger places
60665     IF VAL(VL.SS$(F.SS%))=0 THEN
VL.SS$(F.SS%)=LEFT$(MID$(BLNK.SS$,1,NUMINT%-
```

```
1)+"0."+STRING$(NUMDEC%,"0"),LE.SS%(F.SS%)): GOTO 60690
' If no initial value
60670        DEC.VL%=INSTR(VL.SS$(F.SS%),".") : IF
DEC.VL%=0 THEN DEC.VL%=LE.SS%(F.SS%)+1
' Position of decimal point in data
60675
VL.SS$(F.SS%)=LEFT$(RIGHT$(MID$(BLNK.SS$,1,NUMINT%)+LEFT$(V
L.SS$(F.SS%),DEC.VL%-
1),NUMINT%)+"."+MID$(VL.SS$(F.SS%),DEC.VL%+1)+STRING$(NUMDE
C%,"0"),LE.SS%(F.SS%))
60680 '
60685 '
60690 GOSUB 62310                ' Display Contents Of This
Field
60695 NEXT F.SS%
60700 RETURN
60705 '
60710 '
60715 '
**********************************************
60720 '          *** Accept Input Data For The Current
Field ***
60725 '
**********************************************
60730 '
60740 IF TY.SS$(F.SS%)<>"N" THEN A.SS%=1 : GOTO  60790
60750     NEWNUM%=-1:NUMED.SS%=0      'Set Flags for num
fld
60760     DECPOS%=INSTR(PIC.SS$(F.SS%),".") : IF DECPOS%=0
THEN DECPOS%=LE.SS%(F.SS%)+1
60770     A.SS%=DECPOS%-1
60780 '                                    ' Look for non-
edit characters
60790 WHILE INSTR("ULX#98",MID$(PIC.SS$(F.SS%),A.SS%,1))=0:
A.SS%=A.SS%+1: WEND
60800 CURCOL%=LO.SS%(F.SS%,2)+A.SS%-1     ' Find cursor
position on screen
60810 LOCATE LO.SS%(F.SS%,1),CURCOL%,1    ' Starting
position in this field
60820 '
60830 FLDLST.SS% = F.SS%                 ' Reset field
indicator
60840 EXFLD.SS%=0                        ' Initializei flag
to exit this field
60850 WHILE NOT EXFLD.SS%                ' Loop while still
editing this field
60860   X.SS$=INKEY$ : IF X.SS$="" THEN 60860  ' Wait for
next keyboard character
60870   IF ERR.MSG% THEN ERR.MSG%=0 : COLOR 7,0:LOCATE
25,1,0:PRINT STRING$(79," ");
' Erase old message line.
```

```
60880   IF LEN(X.SS$)>1 OR
INSTR(CHR$(8)+CHR$(13)+CHR$(27),X.SS$)<>0 THEN
X.SS$=RIGHT$(X.SS$,1) ELSE 60960                    '
Extented code key pressed?
60885       ON INSTR(";<=>?@ABCD",X.SS$) GOSUB
61100,61100,61100,61100,61100,61100,61100,61100,61100,61100
' This is a DUMMY statement.  It traps the Function Keys
(F1 - F10).  It is here to make user modifications simpler.
60887               ' For the above line to be active, you
need to set all Function Key values to null. i.e.-  KEY
1,"" ; KEY 2,"" etc.
60890       IF TY.SS$(F.SS%)<>"N" THEN 60920        'If
numeric, need special test
60900          IF INSTR("GO",X.SS$)<>0 THEN 61030  'Codes
not valid for numeric
60910          IF INSTR("RKM"+CHR$(8),X.SS$)<>0 THEN
NUMED.SS%=-1:NEWNUM%=0
60920       IF INSTR(EXITCHR.SS$,X.SS$)<>0 THEN EXFLD.SS%=-
1:EXSCR.SS%=-1:GOTO 61040  'Check CODE to EXIT SCREEN
60930       ON INSTR("MKHPGRSO"+CHR$(8)+CHR$(13),X.SS$)
GOSUB
61110,61140,61210,61260,61650,61440,61300,61600,61140,61260
: GOTO 61020
60940       GOTO 61030              ' Invalid extended code
key pressed
60950 '
60960   IF TY.SS$(F.SS%)="N" AND X.SS$="." THEN
NEWNUM%=0:NUMED.SS%=-1: GOTO 61010
60970   IF ASC(X.SS$)<32 OR ASC(X.SS$)>126 THEN 61030   '
Invalid characters
60980   GOSUB 61750 :IF ERR.MSG% THEN 61030 ' Non-spec char
entered: Test entry
60990   GOSUB 62230                             ' Add char to
this field
61000   GOSUB 62310                             ' Print new
field
61010   GOSUB 62380                             ' Move cursor
to next position
61020   IF ERR.MSG% THEN  60740                 ' If error re-
edit this field
61030 WEND
61040 LASTCHR.SS$=X.SS$                            ' Set last
character indicator
61050 RETURN                                    ' Exit from
editing this field
61060 '
61070
'*************************************************************
*************
61080 '****  THE FOLLOWING SUBROUTINES HANDLE EXTENDED
KEYBOARD COMMANDS  ****
```

```
61090
'*************************************************************
*************
61095 ' ***  DUMMY Subroutine for Function Keys (F1 - F10)
61096 ' ***  This is here for user modifications   ***
61100 RETURN
61105 '*** Cursor right ***
61110    GOSUB 62380            ' Move cursor to next
position
61120    RETURN
61130 '*** Cursor left or backspace ***
61140    IF CURCOL%=LO.SS%(F.SS%,2) THEN 61210       'Goto
prior field
61150    IF TY.SS$(F.SS%)="N" AND INSTR(" +-
",MID$(VL.SS$(F.SS%),A.SS%,1))<>0 THEN RETURN
61160    CURCOL%=CURCOL%-1:A.SS%=A.SS%-1
'Pos of char in field
61170    IF INSTR("ULX#89",MID$(PIC.SS$(F.SS%),A.SS%,1))=0
THEN 61140 'Spec char?
61180    LOCATE LO.SS%(F.SS%,1),CURCOL%,1
61190    RETURN
61200 '*** Cursor up ***      Move to next field left
61210    GOSUB 62500                    ' Edit check field
data before leaving
61220    IF ERR.MSG% THEN RETURN        ' Error found
61225    EXFLD.SS%=-1                   ' Set Flag to Exit
this Field
61230    IF F.SS%>1 THEN F.SS%=F.SS%-1 ELSE
F.SS%=NUMFLDS.SS%                              ' Goto
prior fld
61240    RETURN
61250 '*** Cursor down or carriage return - Advance to next
field ***
61260    GOSUB 62500                ' Edit check field
data before leaving
61270    IF ERR.MSG% THEN  RETURN   ' Don't leave field
if error was found
61275    EXFLD.SS%=-1               ' Set Flag to Exit
this Field
61277    IF F.SS%=NUMFLDS.SS% AND
INSTR(EXITCHR.SS$,CHR$(127))<>0 THEN EXSCR.SS%=-1  'Test to
leave screen after last field
61280    IF F.SS%<NUMFLDS.SS% THEN F.SS%=F.SS%+1 ELSE
F.SS%=1                          ' Increment fld num to
next fld
61290    RETURN
61300 ' **** Del key pressed ******
61310 '              ' Start Del routine for Numeric fld on
left of decimal pt
```

```
61320 IF TY.SS$(F.SS%)="N" AND A.SS%<DECPOS% THEN
MID$(VL.SS$(F.SS%),1)=" "+LEFT$(VL.SS$(F.SS%),A.SS%-
1)+RIGHT$(VL.SS$(F.SS%),LE.SS%(F.SS%)-A.SS%): GOTO 61420
61330 '               ' Start Del routine for Numeric fld on
right of decimal pt
61340 IF TY.SS$(F.SS%)="N" THEN
MID$(VL.SS$(F.SS%),1)=LEFT$(VL.SS$(F.SS%),A.SS%-
1)+MID$(VL.SS$(F.SS%),A.SS%+1,LE.SS%(F.SS%)-A.SS%)+"0" :
GOTO 61420
61350 '               ' Start Del routine for fld w/o non-
edit chr
61360 IF SPECCHR.SS%(F.SS%)=0 THEN
MID$(VL.SS$(F.SS%),1)=LEFT$(VL.SS$(F.SS%),A.SS%-
1)+MID$(VL.SS$(F.SS%),A.SS%+1,LE.SS%(F.SS%)-A.SS%)+" " :
GOTO 61420
61370 '
61380 CNT.SS%=0                    ' Start del routine for
fld with non-edit chr
61390 WHILE
INSTR("ULX#89",MID$(PIC.SS$(F.SS%),A.SS%+1+CNT.SS%,1))<>0
AND CNT.SS%<LE.SS%(F.SS%)-A.SS% : CNT.SS%=CNT.SS%+1 :  WEND
' Count until next non-edit chr
61400 VL.SS$(F.SS%)=LEFT$(VL.SS$(F.SS%),A.SS%-
1)+MID$(VL.SS$(F.SS%),A.SS%+1,CNT.SS%)+"
"+RIGHT$(VL.SS$(F.SS%),LE.SS%(F.SS%)-A.SS%-CNT.SS%)     '
New value for field
61410 '
61420 CURCOL%=CURCOL%-1:A.SS%=A.SS%-1:GOSUB 62340:GOSUB
62380 'Print fld; Set cursor
61430 RETURN
61440 ' ***** Ins key pressed ***
61450 '               ' Start Ins routine for numeric field on
leftside of dec pt
61460 IF TY.SS$(F.SS%)="N" AND A.SS%<DECPOS% THEN
MID$(VL.SS$(F.SS%),1)=MID$(VL.SS$(F.SS%),2,A.SS%-
1)+"0"+RIGHT$(VL.SS$(F.SS%),LE.SS%(F.SS%)-A.SS%) : GOTO
61580
61470 '               ' Start Ins routine for numeric field on
rightside of dec pt
61480 IF TY.SS$(F.SS%)="N"  THEN
VL.SS$(F.SS%)=LEFT$(VL.SS$(F.SS%),A.SS%-
1)+"0"+MID$(VL.SS$(F.SS%),A.SS%,LE.SS%(F.SS%)-A.SS%) : GOTO
61580
61490 '               ' Start Ins routine for non-numeric w/o
non-edit characters
61500 IF SPECCHR.SS%(F.SS%)=0 THEN
VL.SS$(F.SS%)=LEFT$(VL.SS$(F.SS%),A.SS%-1)+"
"+MID$(VL.SS$(F.SS%),A.SS%,LE.SS%(F.SS%)-A.SS%) : GOTO
61580
61510 '               ' Start Ins routine for non-numeric with
non-edit characters
```

```
61520 NEWVL$=LEFT$(VL.SS$(F.SS%),A.SS%-1)+" " :
NEXTCHR$=MID$(VL.SS$(F.SS%),A.SS%,1)
61530 FOR I%=A.SS%+1 TO LE.SS%(F.SS%)
61540   X.SS$=MID$(PIC.SS$(F.SS%),I%,1): IF
INSTR("ULX#89",X.SS$)=0 THEN NEWVL$=NEWVL$+X.SS$ : GOTO
61570
61550   NEWVL$=NEWVL$+NEXTCHR$ :
NEXTCHR$=MID$(VL.SS$(F.SS%),I%,1)
61560 NEXT I%
61570
VL.SS$(F.SS%)=NEWVL$+MID$(VL.SS$(F.SS%),I%+1,LE.SS%(F.SS%))
61580 CURCOL%=CURCOL%-1:A.SS%=A.SS%-1:GOSUB 62310:GOSUB
62380 ' Print fld; Set cursor
61590 RETURN
61600 ' ***** END key pressed *****
61610 CURCOL%=LO.SS%(F.SS%,2)+LE.SS%(F.SS%)-1  :
A.SS%=LE.SS%(F.SS%)
61620 WHILE INSTR("ULX#89",MID$(PIC.SS$(F.SS%),A.SS%,1))=0
:A.SS%=A.SS%-1:CURCOL%=CURCOL%-1: WEND       ' Look for
special protected characters
61630 LOCATE LO.SS%(F.SS%,1),CURCOL%,1       ' Starting
position in this field
61640 RETURN
61650 ' **** HOME key pressed ****  put cursor at beginning
of field
61660 A.SS%=1                        'Find cursor position for
this field
61670 WHILE INSTR("ULX#89",MID$(PIC.SS$(F.SS%),A.SS%,1))=0
: A.SS%=A.SS%+1 : WEND     ' Look for special protected
characters
61680 CURCOL%=LO.SS%(F.SS%,2)+A.SS%-1
61690 LOCATE LO.SS%(F.SS%,1),CURCOL%,1 ' Starting position
in this field
61700 RETURN
61710 '
61715 '
************************************************
61720 '        ****  ROUTINE TO EDIT-TEST CHARACTER ENTRY
****
61730 '
************************************************
61740 '***Check for special character type conversion***
61750 ON INSTR("NDMY",TY.SS$(F.SS%)) GOTO
61790,62100,62000,61950
61760 ON INSTR("ULX#89",MID$(PIC.SS$(F.SS%),A.SS%,1)) GOTO
61890,61920,62130,61800,62060,62100
61770 PRINT "EDIT PICTURE TYPE
";MID$(PIC.SS$(F.SS%),A.SS%,1);" NOT FOUND" : STOP
61780 '
61790 '*** Numeric values; "."; "-"; "+"; " "
```

```
61800 ' NOTE: The decimal point is trapped in the"Accept
input data" routine
61810 IF X.SS$>="0" AND X.SS$<="9" THEN RETURN
61820 IF X.SS$<>"+" AND X.SS$<>"-" AND X.SS$<>" " THEN
61850  ' Is this a + or - sign?
61830 IF TY.SS$(F.SS%)="C" THEN RETURN ' +,-," " allowed
anywhere in C type
61840    IF LEFT$(VL.SS$(F.SS%),A.SS%-1)=SPACE$(A.SS%-1) OR
A.SS%=1 THEN RETURN
61841 '      " "  "+" and "-" sign only allowed in
beginning of number
61850    MSG.SS$=" Only numeric values can be entered here.
Please re-enter. "
61860    GOSUB 62180 : RETURN        ' Print error message;
Exit
61870 '
61880 '*** Upper case or any other character***
61890 IF ASC(X.SS$)>96 AND ASC(X.SS$)<123 THEN
X.SS$=CHR$(ASC(X.SS$)-32)
61900 GOTO 62130
61910 '*** Lower case or any other character***
61920 IF ASC(X.SS$)>65 AND ASC(X.SS$)<91 THEN
X.SS$=CHR$(ASC(X.SS$)+32)
61930 GOTO 62130
61940 '*** Y/N answer only***
61950 IF X.SS$="Y" OR X.SS$="y" THEN X.SS$="Y" : GOTO 62130
61960 IF X.SS$="N" OR X.SS$="n" THEN X.SS$="N" : GOTO 62130
61970    MSG.SS$=" Only 'Y' or 'N' can be entered here.
Please re-enter. "
61980    GOTO 62180        ' Print error message
61990 '*** M/F answer only***
62000 IF X.SS$="M" OR X.SS$="m" THEN X.SS$="M" : GOTO 62130
62010 IF X.SS$="F" OR X.SS$="f" THEN X.SS$="F" : GOTO 62130
62020    MSG.SS$=" Only 'M' or 'F' can be entered here.
Please re-enter. "
62030    GOTO 62180      ' Print error message
62040 '
62050 ' *** Numeric values only ****
62060 IF (ASC(X.SS$)>47 AND ASC(X.SS$)<58) THEN GOTO 62130
62070    MSG.SS$=" Only numeric values can be entered here.
Please re-enter. "
62080    GOTO 62180        ' Print error message
62090 ' *** Numeric values and " " only ***
62100 IF (ASC(X.SS$)>47 AND ASC(X.SS$)<58) OR X.SS$=" "
THEN GOTO 62130
62110    MSG.SS$=" Only numeric values or blanks can be
entered here. Please re-enter. "
62120    GOTO 62180        ' Print error message
62130 RETURN
62140 '
62150 ' *********************************
```

```
62160 ' **** Print Error Messages ******
62170 ' ********************************
62180   ERR.MSG%=-1 : SOUND 500,SD.SS%*1:LOCATE 25,INT(81-
LEN(MSG.SS$))/2,0 :COLOR 0,7:PRINT MSG.SS$;: LOCATE
LO.SS%(F.SS%,1),CURCOL%,1
62190   WHILE INKEY$<>"" : WEND    ' Clear input buffer
after error
62200 RETURN
62210 '
62220 ' ****************************************
62230 ' **** Add character to current field *****
62240 ' ****************************************
62250 IF TY.SS$(F.SS%)="N" AND NEWNUM% THEN
MID$(VL.SS$(F.SS%),1)=MID$(BLNK.SS$,1,A.SS%-
1)+X.SS$+"."+STRING$(LE.SS%(F.SS%),"0") : NEWNUM%=0 :
RETURN  ' Restart fld - New num
62260 IF TY.SS$(F.SS%)<>"N" OR NUMED.SS%=-1 THEN
MID$(VL.SS$(F.SS%),A.SS%,1)=X.SS$ : RETURN
62270 IF LEFT$(VL.SS$(F.SS%),1)=" " THEN
MID$(VL.SS$(F.SS%),1,A.SS%)=MID$(VL.SS$(F.SS%),2,A.SS%-
1)+X.SS$  ELSE  NUMED.SS%=-1 : GOTO 62260    ' Add
character to left of decimal place
62280 RETURN
62290 '
62300 '*******************************
62310 '*** Print new value of field ***
62320 '*******************************
62330 '
62340 COLOR CL.SS%(F.SS%,1),CL.SS%(F.SS%,2)          ' Set
color for this field
62350 LOCATE LO.SS%(F.SS%,1),LO.SS%(F.SS%,2),0 : PRINT
VL.SS$(F.SS%); ' Print field
62360 RETURN
62365 '
62370 '************************************
62380 '**** Move cursor to new location ****
62390 '************************************
62400 IF TY.SS$(F.SS%)<>"N" OR NUMED.SS%<>0 THEN 62420
62410    IF LEFT$(VL.SS$(F.SS%),1)<>" " THEN NUMED.SS%=-1
ELSE A.SS%=DECPOS%-1:CURCOL%=LO.SS%(F.SS%,2)+A.SS%-1:LOCATE
LO.SS%(F.SS%,1),CURCOL%,1 : RETURN
62415 '
62420 IF A.SS%<LE.SS%(F.SS%) THEN
A.SS%=A.SS%+1:CURCOL%=CURCOL%+1 ELSE GOSUB 61260 : RETURN
' Advance cursor or go to next field
62430     IF INSTR("ULX#89",MID$(PIC.SS$(F.SS%),A.SS%,1))=0
THEN 62420
62440     LOCATE LO.SS%(F.SS%,1),CURCOL%,1
62450 RETURN
62460 '
62470 ' ************************************
```

```
62480 ' ***  Edit check final field result ***
62490 ' ************************************
62500 IF TY.SS$(F.SS%)<>"N" THEN 62540    ' Check numeric
input range
62510     IF VAL(VL.SS$(F.SS%))>RG.SS(F.SS%,2) THEN
MSG.SS$=" The maximum value allowed in this field is
"+STR$(RG.SS(F.SS%,2)): GOSUB 62180 : RETURN  'Print Err
Msg
62520     IF VAL(VL.SS$(F.SS%))<RG.SS(F.SS%,1) THEN
MSG.SS$=" The minimum value allowed in this field is
"+STR$(RG.SS(F.SS%,1)): GOSUB 62180 : RETURN  'Print Err
Msg
62530 '
62540 IF TY.SS$(F.SS%)<>"D" THEN 62720         ' Date edit
check
62550     IF VL.SS$(F.SS%)="  /  /  " THEN 62720
' No check
62560
DT.SS=VAL(RIGHT$(VL.SS$(F.SS%),2)+LEFT$(VL.SS$(F.SS%),2)+MI
D$(VL.SS$(F.SS%),4,2))
62570     IF
INSTR("01,02,03,04,05,06,07,08,09,10,11,12",LEFT$(VL.SS$(F.
SS%),2))<>0 THEN 62590
62580         MSG.SS$="Invalid MONTH in date entered. Please
re-enter.":GOSUB 62180 : RETURN
62590     IF INSTR(MID$(VL.SS$(F.SS%),4,2)," ")=0 AND
VAL(MID$(VL.SS$(F.SS%),4,2))>0 AND
VAL(MID$(VL.SS$(F.SS%),4,2))<32 THEN 62620
62600         MSG.SS$="Invalid DAY in date entered.  Please
re-enter.":GOSUB 62180:RETURN
62610 '
62620     IF RG.SS(F.SS%,1)=0 AND RG.SS(F.SS%,2)=0 THEN
62720  ' No check
62630     IF RG.SS(F.SS%,2)<RG.SS(F.SS%,1) THEN 62680
'Does date cross century?
62640         IF DT.SS>=RG.SS(F.SS%,1) AND
DT.SS<=RG.SS(F.SS%,2) THEN 62720
62650             D1.SS$=STR$(RG.SS(F.SS%,1)) :
D2.SS$=STR$(RG.SS(F.SS%,2))
62660             MSG.SS$="The date should be between
"+MID$(D1.SS$,4,2)+"/"+RIGHT$(D1.SS$,2)+"/"+MID$(D1.SS$,2,2
)+" and
"+MID$(D2.SS$,4,2)+"/"+RIGHT$(D2.SS$,2)+"/"+MID$(D2.SS$,2,2
) :GOSUB 62180 : RETURN
62670 '
62680     IF DT.SS>=RG.SS(F.SS%,2) OR DT.SS<=RG.SS(F.SS%,1)
THEN 62720
62690             D1.SS$=STR$(RG.SS(F.SS%,2)) :
D2.SS$=STR$(RG.SS(F.SS%,1))
62700             MSG.SS$="The date should be between
"+MID$(D1.SS$,4,2)+"/"+RIGHT$(D1.SS$,2)+"/"+MID$(D1.SS$,2,2
```

```
)+" and
"+MID$(D2.SS$,4,2)+"/"+RIGHT$(D2.SS$,2)+"/"+MID$(D2.SS$,2,2
) :GOSUB 62180 : RETURN
62710 '
62720 RETURN
62860 ' ************************************************
62870 ' ** Test If Monochrome or Color/Graphics Monitor **
62880 ' ************************************************
62890 DEF SEG=&H40
62900 MONO.SS=(PEEK(&H10) AND &H30)=&H30
62910 IF MONO.SS THEN SCRNSEG.SS%=&HB000 ELSE
SCRNSEG.SS%=&HB800
62920 RETURN
62930 ' ******************************
62940 ' **  Error Handling Routine  **
62950 ' ******************************
62960 ' This routine is used mostly to turn the screen
display back on if
62970 ' a serious problem (such as the screen image file
not found) occurs.
62980 CLS:OUT &H3D8,&H29        ' Turn screen display BACK
ON (if off).
62990                        'The following checks for disk
file errors
63000 IF NOT (ERR=53 OR ERR=57 OR ERR=66 OR (ERR>70 AND
ERR<77)) THEN 63030
63010    LOCATE 10,1:PRINT"There is a problem finding a
file on the disk.   Error code="+STR$(ERR)
63020    LOCATE 11,1:PRINT"(HINT:Check disk for .SCR
file)":PRINT
63030 ON ERROR GOTO 0       ' Reset BASIC ERROR handling
```

```
6660 '********** NAVIGATION SUBROUTINE **********
6661 SCREEN 0,1,1,0:COLOR 15,0,4:SCREEN 0,1,0,0:COLOR
15,0,4:CLS:CONTROL1=0:TOTALDIST=0:PRINT
"~K={BACK},KEYFIX,NOESC,NOMOVE/"
6662 PRINT "~W=LOADOPT/":SCREEN ,,3,0:INPUT Z:IF Z=2 THEN
CHAIN "FILEHALF",,ALL
6663 SCREEN 0,1,0,0:COLOR 15,0,4:CLS:PRINT:ON ERROR GOTO
8600
6664 PRINT TAB(20) "INPUT FORMAT => _____DEG  _____MIN
     _____SEC"
6665 PRINT
6666 PRINT TAB(20) "SELECT THE GEOGRAPHIC DATA FOR YOUR
LAT/LONG POINTS"
6667 PRINT
6668 LOCATE 8,25:INPUT "LATITUDE IS N OR S  ";GEOLAT$
'South is negative
6669 LOCATE 10,25:INPUT "LONGITUDE IS W OR E  ";GEOLONG$
'East is negative
6670 LOCATE 12,25:INPUT "MAGNETIC VARIATION IS W OR E
";MVARTYPE$
6671 IF GEOLAT$="n" THEN GEOLAT$="N"
6672 IF GEOLAT$="s" THEN GEOLAT$="S"
6673 IF GEOLONG$="w" THEN GEOLONG$="W"
6674 IF GEOLONG$="e" THEN GEOLONG$="E"
6675 IF MVARTYPE$="w" THEN MVARTYPE$="W"
6677 IF MVARTYPE$="e" THEN MVARTYPE$="E"
6678 LOCATE 15,20:INPUT "ENTER THE NUMBER OF NAVIGATION
POINTS";N:CLS
6679 LOCATE 2,25:PRINT "NAVIGATION POINT DATA INPUT":LOCATE
2,74:PRINT "PAGE 1"
6680 LOCATE 4,1
6681 FOR I=1 TO N
6682 IF I=10 THEN SCREEN ,,1,0                    'WRITE TO
PAGE 1
6683 IF I=10 THEN CLS:IF I=10 THEN LOCATE 2,74:IF I=10 THEN
PRINT "PAGE 2"
6684 IF I=10 THEN LOCATE 4,1
6685  PRINT TAB(5) "POINT #";I;TAB(19) "___ DEG     __ MIN
__ SEC ";:COLOR 14:PRINT GEOLAT$:COLOR 15
6686  PRINT TAB(19) "___ DEG      __ MIN      __ SEC ";:COLOR
14:PRINT GEOLONG$;:COLOR 15:PRINT "      MVAR=  __ DEG
";:COLOR 14:PRINT MVARTYPE$:COLOR 15
6687 NEXT I
6688 IF N>9 THEN SCREEN ,,0,0                     'WRITE TO
PAGE 0
6689 COLOR 12
6690 J=3
6691 FOR I=1 TO N
6692 IF I<=9 THEN K=I ELSE K=I-9
6693 IF I>9 THEN SCREEN ,,1,1                     'DISPLAY and
WRITE TO PAGE 1
```

```
6694 IF I>9 THEN J=K+2
6695 NAVDATA(I,1)=I
6696 LOCATE K+J,18,1,0,7:INPUT "
",LATDEG:NAVDATA(I,2)=LATDEG
6697 LOCATE K+J,30,1,0,7:INPUT "
",LATMIN:NAVDATA(I,3)=LATMIN/60
6698 LOCATE K+J,41,1,0,7:INPUT "
",LATSEC:NAVDATA(I,4)=LATSEC/6000
6699 LOCATE K+J+1,18,1,0,7:INPUT "
",LONGDEG:NAVDATA(I,5)=LONGDEG
6700 LOCATE K+J+1,30,1,0,7:INPUT "
",LONGMIN:NAVDATA(I,6)=LONGMIN/60
6701 LOCATE K+J+1,41,1,0,7:INPUT "
",LONGSEC:NAVDATA(I,7)=LONGSEC/6000
6702 LOCATE K+J+1,62,1,0,7:INPUT " ",MVAR:IF
(MVARTYPE$="E") OR (MVARTYPE$="e") THEN
NAVDATA(I,8)=MVAR*(-1) ELSE NAVDATA(I,8)=MVAR
6703 J=J+1
6705 WAYPTS(I,1)=I
6707 WAYPTS(I,2)=LATDEG+LATMIN/100+LATSEC/10000
6709 WAYPTS(I,3)=LONGDEG+LONGMIN/100+LONGSEC/10000
6720 NEXT I
6730 IF N>9 THEN SCREEN ,,0,0                    'DISPLAY and
WRITE TO PAGE 0
6740 GOSUB 7660:IF SELECTFLAG%=2 THEN ERASE
NAVDATA,COORD,WAYPTS:CHAIN "TOPHALF",103,ALL
6745 '
6750 'LOCATE 1,1:SCREEN ,,2,2            'SCREENS 0 and 1
ARE SAVED FOR LAT/LONG
6760 ' NOTE:                              'SCREEN 2 IS NOW
THE APAGE, VPAGE
6770 ' EDIT INPUT DATA USING WINDOWS
6790 '
6800 ' COORDINATE COMPUTATIONS
6810 '
6820 COLOR 15,1
6840 COORD(1,1)=1:COORD(1,2)=0:COORD(1,3)=0
6850 FOR I=1 TO N-1
6860  L1=NAVDATA(I,2)+NAVDATA(I,3)+NAVDATA(I,4)
6870  IF (GEOLAT$="S") OR (GEOLAT$="s") THEN L1=L1*(-1)
6880  L2=NAVDATA(I+1,2)+NAVDATA(I+1,3)+NAVDATA(I+1,4)
6890  IF (GEOLAT$="S") OR (GEOLAT$="s") THEN L2=L2*(-1)
6900  LAMBDA1=NAVDATA(I,5)+NAVDATA(I,6)+NAVDATA(I,7)
6910  IF (GEOLONG$="E") OR (GEOLONG$="e") THEN
LAMBDA1=LAMBDA1*(-1)
6920  LAMBDA2=NAVDATA(I+1,5)+NAVDATA(I+1,6)+NAVDATA(I+1,7)
6930  IF (GEOLONG$="E") OR (GEOLONG$="e") THEN
LAMBDA2=LAMBDA2*(-1)
6940  GOSUB 7040
6950  COORD(I+1,1)=I+1
6960  COORD(I+1,2)=D
```

```
6970  COORD(I+1,3)=HDG+NAVDATA(I+1,8)
'COMPUTE MAGNETIC HDG
6975 IF COORD(I+1,3)<0 THEN COORD(I+1,3)=360+COORD(I+1,3)
'DISPLAY MAGHDG
6980 NEXT I
6981 FOR I=2 TO N
6982  TOTALDIST=TOTALDIST+COORD(I,2)
6983 NEXT I
6984 'CRUSDATA(26)=TOTALDIST
6990 PRINT "~K={BACK}/"
7000 ' PRINT RESULTS
7012 GOSUB 8000
7015 GOSUB 8200
7040 '********* NAVIGATION COMPUTATION SUBROUTINE
**********
7050 '
7060 PI=3.141593
7070 L1=L1*PI/180
7080 L2=L2*PI/180
7090 LAMBDA1=LAMBDA1*PI/180
7100 LAMBDA2=LAMBDA2*PI/180
7110 DEF FNARCCOS(X)=1.570796-ATN(X/SQR(1-X^2))
7120 X=(SIN(L1)*SIN(L2)+COS(L1)*COS(L2)*COS(LAMBDA2-
LAMBDA1))
7130 D=60*FNARCCOS(X)*180/PI
7140 Y=(D/60)*PI/180
7150 Z=(SIN(L2)-(SIN(L1)*COS(Y)))/(SIN(Y)*COS(L1))
7155 IF Z>1 THEN Z=1
7160 HDG=FNARCCOS(Z)*180/PI
7170 IF (SIN(LAMBDA2-LAMBDA1)>=0) THEN HDG=360-HDG
7180 RETURN
7190 STOP
7200 '********** SUBROUTINE DBLBOX **********
7210 ' TOP = ROW NUMBER OF UPPER LEFT CORNER
7220 ' LEFT = COLUMN OF UPPER LEFT CORNER
7230 ' WIDTZ = WIDTH OF BOX
7240 ' HEIGHT = HEIGHT OF BOX
7250 '
7260 BOTTOM = TOP + HEIGHT
7270 RIGHT = LEFT + WIDTZ
7280 '
7290 ' LOCATE AND PRINT THE CORNERS OF THE BOX
7300 '
7310 LOCATE  TOP, LEFT
7320 PRINT  CHR$(201)          'PRINTs upper left corner
7330 LOCATE  TOP, RIGHT
7340 PRINT CHR$(187)           'PRINTs upper right corner
7350 LOCATE BOTTOM, LEFT
7360 PRINT CHR$(200)           'PRINTs lower left corner
7370 LOCATE BOTTOM, RIGHT
7380 PRINT CHR$(188)           'PRINTs lower right corner
```

```
7390 '
7400 ' DRAW THE TOP, BOTTOM, AND SIDES OF THE BOX
7410 '
7420 FOR COL = (LEFT + 1) TO (RIGHT - 1)        'top and
bottom
7430  LOCATE TOP, COL
7440  PRINT CHR$(205)
7450  LOCATE BOTTOM, COL
7460  PRINT CHR$(205)
7470 NEXT COL
7480 FOR ROW = (TOP + 1) TO (BOTTOM - 1)        'left and
right sides
7490  LOCATE ROW, LEFT
7500  PRINT CHR$(186)
7510  LOCATE ROW, RIGHT
7520  PRINT CHR$(186)
7530 NEXT ROW
7540 RETURN
7550 '
7560 '********** PRINT SCREEN ASSEMBLY LANGUAGE SUBROUTINE
**********
7580 A%(0)=&HCD55
7590 A%(1)=&H5D05
7600 A%(2)=&H90CB
7610 SUBRT = VARPTR(A%(0))
7620 CALL SUBRT
7640 RETURN
7650 '
7660 '********** LAT/LONG POINTS DISPLAY SUBROUTINE
**********
7662 LOCATE 1,1,0:PRINT "~W=EDITSEL/":SCREEN ,,3,0:INPUT
"",DISP:SCREEN ,,0,0
7663 IF DISP=4 THEN CHAIN "ZEROOUT",,ALL
7664 IF DISP=3 THEN RETURN
7666 IF DISP=2 THEN LOCATE 1,1:GOTO 7711
7668 IF DISP=1 THEN GOSUB 7800
7670 GOTO 7660
7711 PRINT "~W=FILING,NOWAIT/":LOCATE 1,1:INPUT "ENTER
DRIVE:FILENAME    => ";A$:OPEN A$ FOR OUTPUT AS #1:WRITE
#1,N
7712 FOR I=1 TO N:FOR J=1 TO 8:WRITE #1,NAVDATA(I,J):NEXT
J:NEXT I
7714 CLOSE #1:PRINT "~C=LAST/":LOCATE 1,1:PRINT "
"
7715 B$=A$+".WPT"
7716 OPEN B$ FOR OUTPUT AS #1:WRITE #1, N: FOR I=1 TO N:FOR
J=1 TO 3: WRITE #1, WAYPTS(I,J):NEXT J:NEXT I:CLOSE #1
7718 C$=A$+".CHA":OPEN C$ FOR OUTPUT AS #1:WRITE
#1,GEOLAT$,GEOLONG$,MVARTYPE$:CLOSE #1:GOTO 7660
7720 RETURN
7800 '********** LAT/LONG EDITING SUBROUTINE **********
```

```
7805 LOCATE 1,1,1,0,7:PRINT "EDIT POINT # ";
7810 LOCATE 1,3:PRINT "~W=EDITPTS/";:INPUT B
7812 IF (DISP=1) AND (B=0) GOTO 7670
7814 IF (DISP=1) AND (B<>0) THEN PTNUM=B
7820 LOCATE 1,1,0
7830 IF PTNUM>9 THEN SCREEN 0,1,1,1 ELSE SCREEN 0,1,0,0
7840 ROW=0
7850 FOR I=1 TO PTNUM
7860  IF PTNUM=I THEN LOCATE PTNUM+I+2,19
7870  ROW=PTNUM+I+2
7880 NEXT I
7890 PRINT "___";:LOCATE ROW,31:PRINT "__";:LOCATE
ROW,42:PRINT "__";
7900 LOCATE ROW+1,19:PRINT "___";:LOCATE ROW+1,31:PRINT
"__";:LOCATE ROW+1,42:PRINT "__";:LOCATE ROW+1,63:PRINT
"__";
7910 LOCATE ROW,18:INPUT " ",LATDEG:NAVDATA(PTNUM,2)=LATDEG
7920 LOCATE ROW,30:INPUT "
",LATMIN:NAVDATA(PTNUM,3)=LATMIN/60
7930 LOCATE ROW,41:INPUT "
",LATSEC:NAVDATA(PTNUM,4)=LATSEC/6000
7940 LOCATE ROW+1,18:INPUT "
",LONGDEG:NAVDATA(PTNUM,5)=LONGDEG
7950 LOCATE ROW+1,30:INPUT "
",LONGMIN:NAVDATA(PTNUM,6)=LONGMIN/60
7960 LOCATE ROW+1,41:INPUT "
",LONGSEC:NAVDATA(PTNUM,7)=LONGSEC/6000
7970 LOCATE ROW+1,62:INPUT " ",MVAR:IF (MVARTYPE$="E") OR
(MVARTYPE$="e") THEN NAVDATA(PTNUM,8)=MVAR*-1 ELSE
NAVDATA(PTNUM,8)=MVAR
7972 WAYPTS(PTNUM,2)=LATDEG+LATMIN/100+LATSEC/10000
7974 WAYPTS(PTNUM,3)=LONGDEG+LONGMIN/100+LONGSEC/10000
7976 LOCATE 1,1,0:PRINT "                                "
7980 RETURN
7990 END
8000 '********** COORDINATE COMPUTATION PRINTOUT SUBROUTINE
**********
8010 '
8020 CLS
8030 LOCATE 3,20:PRINT "TO POINT #     DISTANCE (nm)
MAG HDG (deg)"
8040 LOCATE 4,20:PRINT
"================================================"
8050 FOR I=1 TO N
8060  LOCATE I+4,22:PRINT COORD(I,1)
8070  LOCATE I+4,33:PRINT USING "#####.#";COORD(I,2)
8080  LOCATE I+4,51:PRINT USING "#####";COORD(I,3)
8090 NEXT I
8095 LOCATE 16,20:PRINT "NAV ROUTE DISTANCE = ";:PRINT
USING "####.#";TOTALDIST;:PRINT " nm"
8100 TOP=2:LEFT=14:WIDTZ=55:HEIGHT=20:GOSUB 7200
```

```
8105 IF OUTPUT=1 THEN GOSUB 7560:LPRINT CHR$(12)
'AUTO PRINTSCREEN
8110 LOCATE 24,27,0:PRINT "PRESS ANY KEY TO CONTINUE"
8120 ANS$=INKEY$:IF ANS$="" THEN 8120
8130 RETURN
8200 '********** LOW LEVEL NAV AND PERFORMANCE SUBROUTINE
**********
8210 HCRUISE=LLALT:ALT=LLALT:HDWIND=0
8225 CHAIN "TOPHALF",3053,ALL                    'CRUISE
PERFORMANCE
8230
A1=WAYPTS(1,2):A2=WAYPTS(1,3):A3=WAYPTS(2,2):A4=WAYPTS(2,3)
8235
A5=WAYPTS(3,2):A6=WAYPTS(3,3):A7=WAYPTS(4,2):A8=WAYPTS(4,3)
8240
A9=WAYPTS(5,2):A10=WAYPTS(5,3):A11=WAYPTS(6,2):A12=WAYPTS(6
,3)
8245
A13=WAYPTS(7,2):A14=WAYPTS(7,3):A15=WAYPTS(8,2):A16=WAYPTS(
8,3)
8250 A17=WAYPTS(9,2):A18=WAYPTS(9,3)
8255
B1=NAVDATA(1,8):B2=NAVDATA(2,8):B3=NAVDATA(3,8):B4=NAVDATA(
4,8)
8260
B5=NAVDATA(5,8):B6=NAVDATA(6,8):B7=NAVDATA(7,8):B8=NAVDATA(
8,8)
8265 B9=NAVDATA(9,8)
8270
C1=COORD(2,3):C2=COORD(2,2):C3=COORD(3,3):C4=COORD(3,2)
8275
C5=COORD(4,3):C6=COORD(4,2):C7=COORD(5,3):C8=COORD(5,2)
8280
C9=COORD(6,3):C10=COORD(6,2):C11=COORD(7,3):C12=COORD(7,2)
8285
C13=COORD(8,3):C14=COORD(8,2):C15=COORD(9,3):C16=COORD(9,2)
8290 LT2=C2/LLKGS                                 'LEG TIME
IN HOURS
8295 D1=FIX(LT2)
8300 D2=(LT2-D1)*(60)
8305 D3=D1
8310 D4=D2
8315 LT3=C4/LLKGS
8320 D5=FIX(LT3)
8325 D6=(LT3-D5)*(60)
8335 D7=D3+D5:D8=D4+D6
8340 IF D8>=60 THEN D8=D8-60:D7=D7+1
8345 LT4=C6/LLKGS
8350 D9=FIX(LT4)
8355 D10=(LT4-D9)*(60)
8365 D11=D7+D9:D12=D8+D10
```

```
8370 IF D12>=60 THEN D12=D12-60:D11=D11+1
8375 LT5=C8/LLKGS
8380 D13=FIX(LT5)
8385 D14=(LT5-D13)*(60)
8395 D15=D11+D13:D16=D12+D14
8400 IF D16>=60 THEN D16=D16-60:D15=D15+1
8405 LT6=C10/LLKGS
8410 D17=FIX(LT6)
8415 D18=(LT6-D17)*(60)
8425 D19=D15+D17:D20=D16+D18
8430 IF D20>=60 THEN D20=D20-60:D19=D19+1
8435 LT7=C12/LLKGS:D21=FIX(LT7):D22=(LT7-D21)*(60)
8445 D23=D19+D21:D24=D20+D22
8450 IF D24>=60 THEN D24=D24-60:D23=D23+1
8455 LT8=C14/LLKGS:D25=FIX(LT8):D26=(LT8-D25)*(60)
8465 D27=D23+D25:D28=D24+D26
8470 IF D28>=60 THEN D28=D28-60:D27=D27+1
8475 LT9=C16/LLKGS:D29=FIX(LT9):D30=(LT9-D29)*(60)
8485 D31=D27+D29:D32=D28+D30
8490 IF D32>=60 THEN D32=D32-60:D31=D31+1
8495
E2=LT2*WF:E4=LT3*WF:E6=LT4*WF:E8=LT5*WF:E10=LT6*WF:E12=LT7*
WF:E14=LT8*WF
8500
E16=LT9*WF:F3=E2+E4+E6+E8+E10+E12+E14+E16:F2=D32:F1=D31
8505 E1=LLFUEL1:E3=E1-E2:E5=E3-E4:E7=E5-E6:E9=E7-E8:E11=E9-
E10:E13=E11-E12
8510 E15=E13-E14:E17=E15-E16:F4=E17
8512 SCREEN 0,1,0,2:CLS:SCREEN ,,0,0:LOCATE 1,1
8515 SCR.SS%=3:INIT.SS%=-1:GOSUB 60000
8517 IF LABEL=1 THEN LOCATE 1,1:COLOR 15:PRINT "ROUTE:
";ROUTE$
8518 IF OUTPUT=1 THEN GOSUB 7560:LPRINT CHR$(12)
'AUTO PRINTSCREEN
8519 LOCATE 25,1:PRINT "PRESS ANY KEY TO CONTINUE";
8520 ANS$=INKEY$:IF ANS$="" THEN 8520:LOCATE 25,1:PRINT "
";
8525 LOCATE 1,1:PRINT "~W=FINALOPT/":SCREEN ,,3,0:INPUT
"",FINALOPT%:SCREEN ,,0,0
8527 '
8530 IF FINALOPT%=1 THEN LOAD "TOPHALF",R
8532 IF FINALOPT%=2 THEN ERASE COORD,NAVDATA,WAYPTS:CHAIN
"TOPHALF",103,ALL
8534 '
8535 IF FINALOPT%=3 THEN CHAIN "TOPHALF",130,ALL
8600 IF ERR=13 THEN PRINT "~W=TYPEERR/":SCREEN
0,1,3,0:INPUT CONTROL1
8610 IF CONTROL1=1 THEN SCREEN 0,1,0,0:CONTROL1=0:RESUME
8620 ON ERROR GOTO 0
10000 ON ERROR GOTO 62980     ' Error Handling Routine -
Delete
```

```
10010                          ' if it conflicts with your
own
30005 '    Variables Section For B:XCINPUT
30015 VL.SS$(1)=STR$(TEMP): VL.SS$(2)=STR$(PA):
VL.SS$(3)=STR$(TOGW):
30020 VL.SS$(4)=STR$(RLENGTH): VL.SS$(5)=STR$(DC6):
VL.SS$(6)=STR$(DC7):
30025 VL.SS$(7)=STR$(DC8): VL.SS$(8)=STR$(DC9):
VL.SS$(9)=STR$(CMACH):
30030 VL.SS$(10)=STR$(HCRUISE): VL.SS$(11)=STR$(HDWIND):
VL.SS$(12)=STR$(HSD):
30035 VL.SS$(13)=STR$(HED): VL.SS$(14)=STR$(FUELLOAD):
30040 RETURN
30050 '    Assign VL.SS$ Array to the variables
30060
TEMP=VAL(VL.SS$(1)):PA=VAL(VL.SS$(2)):TOGW=VAL(VL.SS$(3)):
30065
RLENGTH=VAL(VL.SS$(4)):DC6=VAL(VL.SS$(5)):DC7=VAL(VL.SS$(6)
):
30070
DC8=VAL(VL.SS$(7)):DC9=VAL(VL.SS$(8)):CMACH=VAL(VL.SS$(9)):
30075
HCRUISE=VAL(VL.SS$(10)):HDWIND=VAL(VL.SS$(11)):HSD=VAL(VL.S
S$(12)):
30080 HED=VAL(VL.SS$(13)):FUELLOAD=VAL(VL.SS$(14)):
30085 RETURN
30095 '     Section To Initialize Variables To Initial
Values
30105 TEMP=0: PA=0: TOGW=31200: RLENGTH=13500: DC6=0:
30110 DC7=0: DC8=0: DC9=0: CMACH=0!: HCRUISE=27000:
30115 HDWIND=0: HSD=27000: HED=0: FUELLOAD=10000:
30120 RETURN
30130 '     ****  List DATA statements & Print DISPLAY Only
Variables  ****
30135 'Lin,Col,Len,Picture,Low Range,High
Range,Foreground,Background,# of Edit
30145 DATA 53,5,6,"N","######",0,120,15,0,0
30150 DATA 53,6,6,"N","######",0,8000,15,0,0
30155 DATA 53,7,6,"N","######",25000,42000,15,0,0
30160 DATA 53,8,6,"N","######",0,99999,15,0,0
30165 DATA 53,9,6,"N","###XXX",0,999,15,0,0
30170 DATA 53,10,6,"N","######",0,999,15,0,0
30175 DATA 53,11,6,"N","######",0,999,15,0,0
30180 DATA 53,12,6,"N","######",0,999,15,0,0
30185 DATA 53,13,6,"N","###.##",0.00,1.20,15,0,1
30190 DATA 53,14,6,"N","######",100,46000,15,0,0
30195 DATA 53,15,6,"N","######",-200,200,15,0,0
30200 DATA 53,16,6,"N","######",100,46000,15,0,0
30205 DATA 53,17,6,"N","######",0,46000,15,0,0
30210 DATA 53,19,6,"N","######",0,20000,15,0,0
30215 RETURN
```

```
30225 '    Screen Display Initialization Statements
30235 NUMFLDS.SS%=14: FILNM.SS$="XCINPUT.SCR":
30240 EXITCHR.SS$=CHR$(27)+CHR$(127)+""
30245 RESTORE 30130
30250 RETURN
30255 '
30260 '    Variables Section For B:LLINPUT
30265 '
30270 VL.SS$(1)=STR$(LLALT): VL.SS$(2)=STR$(LLKGS):
VL.SS$(3)=STR$(LLFUEL1):
30275 RETURN
30280 '
30285 '    Assign VL.SS$ Array to the variables
30290 '
30295
LLALT=VAL(VL.SS$(1)):LLKGS=VAL(VL.SS$(2)):LLFUEL1=VAL(VL.SS
$(3)):
30300 RETURN
30305 '
30310 '     Section To Initialize Variables To Initial
Values
30315 '
30320 LLALT=200: LLKGS=360: LLFUEL1=0:
30325 RETURN
30330 '
30335 '     ****  List DATA statements & Print DISPLAY Only
Variables  ****
30340 'Lin,Col,Len,Picture,Low Range,High
Range,Foreground,Background,# of Edit
30345 '
30350 DATA 48,8,5,"N","#####",100,25000,15,0,0
30355 DATA 48,11,5,"N","#####",0,700,15,0,0
30360 DATA 48,14,5,"N","#####",0,16000,15,0,0
30365 RETURN
30370 '
30375 '    Screen Display Initialization Statements
30380 '
30385 NUMFLDS.SS%=3: FILNM.SS$="LLINPUT.SCR":
30390 EXITCHR.SS$=CHR$(27)+CHR$(127)+""
30395 RESTORE 30335
30400 RETURN
30405 '
30410 '    Variables Section For B:LLJETLOG
30415 '
30420 RETURN
30425 '
30430 '    Assign VL.SS$ Array to the variables
30435 '
30440 RETURN
30445 '
```

```
30450 '     Section To Initialize Variables To Initial
Values
30455 '
30460 RETURN
30465 '
30470 '     ****  List DATA statements & Print DISPLAY Only
Variables  ****
30475 'Lin,Col,Len,Picture,Low Range,High
Range,Foreground,Background,# of Edit
30480 '
30485 COLOR 15,0: LOCATE 4,13: PRINT
LEFT$(GEOLAT$+LEFT$(BLNK.SS$,1-LEN(GEOLAT$)),1);
30490 COLOR 15,0: LOCATE 4,17: PRINT
LEFT$(GEOLONG$+LEFT$(BLNK.SS$,1-LEN(GEOLONG$)),1);
30495 COLOR 15,0: LOCATE 4,32: PRINT
LEFT$(MVARTYPE$+LEFT$(BLNK.SS$,1-LEN(MVARTYPE$)),1);
30500 COLOR 14,1: LOCATE 6,12: PRINT USING "####.####"; A1;
30505 COLOR 14,1: LOCATE 7,12: PRINT USING "####.####"; A2;
30510 COLOR 14,1: LOCATE 8,12: PRINT USING "####.####"; A3;
30515 COLOR 14,1: LOCATE 9,12: PRINT USING "####.####"; A4;
30520 COLOR 14,1: LOCATE 10,12: PRINT USING "####.####";
A5;
30525 COLOR 14,1: LOCATE 11,12: PRINT USING "####.####";
A6;
30530 COLOR 14,1: LOCATE 12,12: PRINT USING "####.####";
A7;
30535 COLOR 14,1: LOCATE 13,12: PRINT USING "####.####";
A8;
30540 COLOR 14,1: LOCATE 14,12: PRINT USING "####.####";
A9;
30545 COLOR 14,1: LOCATE 15,12: PRINT USING "####.####";
A10;
30550 COLOR 14,1: LOCATE 16,12: PRINT USING "####.####";
A11;
30555 COLOR 14,1: LOCATE 17,12: PRINT USING "####.####";
A12;
30560 COLOR 14,1: LOCATE 18,12: PRINT USING "####.####";
A13;
30565 COLOR 14,1: LOCATE 19,12: PRINT USING "####.####";
A14;
30570 COLOR 14,1: LOCATE 20,12: PRINT USING "####.####";
A15;
30575 COLOR 14,1: LOCATE 21,12: PRINT USING "####.####";
A16;
30580 COLOR 14,1: LOCATE 22,12: PRINT USING "####.####";
A17;
30585 COLOR 14,1: LOCATE 23,12: PRINT USING "####.####";
A18;
30590 COLOR 14,1: LOCATE 6,27: PRINT USING "######"; B1;
30595 COLOR 14,1: LOCATE 8,27: PRINT USING "######"; B2;
30600 COLOR 14,1: LOCATE 10,27: PRINT USING "######"; B3;
```

```
30605 COLOR 14,1: LOCATE 12,27: PRINT USING "######"; B4;
30610 COLOR 14,1: LOCATE 14,27: PRINT USING "######"; B5;
30615 COLOR 14,1: LOCATE 16,27: PRINT USING "######"; B6;
30620 COLOR 14,1: LOCATE 18,27: PRINT USING "######"; B7;
30625 COLOR 14,1: LOCATE 20,27: PRINT USING "######"; B8;
30630 COLOR 14,1: LOCATE 22,27: PRINT USING "######"; B9;
30635 COLOR 14,1: LOCATE 8,36: PRINT USING "########"; C1;
30640 COLOR 14,1: LOCATE 9,36: PRINT USING "######.#"; C2;
30645 COLOR 14,1: LOCATE 10,36: PRINT USING "########"; C3;
30650 COLOR 14,1: LOCATE 11,36: PRINT USING "######.#"; C4;
30655 COLOR 14,1: LOCATE 12,36: PRINT USING "########"; C5;
30660 COLOR 14,1: LOCATE 13,36: PRINT USING "######.#"; C6;
30665 COLOR 14,1: LOCATE 14,36: PRINT USING "########"; C7;
30670 COLOR 14,1: LOCATE 15,36: PRINT USING "######.#"; C8;
30675 COLOR 14,1: LOCATE 16,36: PRINT USING "########"; C9;
30680 COLOR 14,1: LOCATE 17,36: PRINT USING "######.#";
C10;
30685 COLOR 14,1: LOCATE 18,36: PRINT USING "########";
C11;
30690 COLOR 14,1: LOCATE 19,36: PRINT USING "######.#";
C12;
30695 COLOR 14,1: LOCATE 20,36: PRINT USING "########";
C13;
30700 COLOR 14,1: LOCATE 21,36: PRINT USING "######.#";
C14;
30705 COLOR 14,1: LOCATE 22,36: PRINT USING "########";
C15;
30710 COLOR 14,1: LOCATE 23,36: PRINT USING "######.#";
C16;
30715 COLOR 14,1: LOCATE 8,47: PRINT USING "#####"; D1;
30720 COLOR 14,1: LOCATE 8,53: PRINT USING "##"; D2;
30725 COLOR 14,1: LOCATE 9,47: PRINT USING "#####"; D3;
30730 COLOR 14,1: LOCATE 9,53: PRINT USING "##"; D4;
30735 COLOR 14,1: LOCATE 10,47: PRINT USING "#####"; D5;
30740 COLOR 14,1: LOCATE 10,53: PRINT USING "##"; D6;
30745 COLOR 14,1: LOCATE 11,47: PRINT USING "#####"; D7;
30750 COLOR 14,1: LOCATE 11,53: PRINT USING "##"; D8;
30755 COLOR 14,1: LOCATE 12,47: PRINT USING "#####"; D9;
30760 COLOR 14,1: LOCATE 12,53: PRINT USING "##"; D10;
30765 COLOR 14,1: LOCATE 13,47: PRINT USING "#####"; D11;
30770 COLOR 14,1: LOCATE 13,53: PRINT USING "##"; D12;
30775 COLOR 14,1: LOCATE 14,47: PRINT USING "#####"; D13;
30780 COLOR 14,1: LOCATE 14,53: PRINT USING "##"; D14;
30785 COLOR 14,1: LOCATE 15,47: PRINT USING "#####"; D15;
30790 COLOR 14,1: LOCATE 15,53: PRINT USING "##"; D16;
30795 COLOR 14,1: LOCATE 16,47: PRINT USING "#####"; D17;
30800 COLOR 14,1: LOCATE 16,53: PRINT USING "##"; D18;
30805 COLOR 14,1: LOCATE 17,47: PRINT USING "#####"; D19;
30810 COLOR 14,1: LOCATE 17,53: PRINT USING "##"; D20;
30815 COLOR 14,1: LOCATE 18,47: PRINT USING "#####"; D21;
30820 COLOR 14,1: LOCATE 18,53: PRINT USING "##"; D22;
```

```
30825 COLOR 14,1: LOCATE 19,47: PRINT USING "#####"; D23;
30830 COLOR 14,1: LOCATE 19,53: PRINT USING "##"; D24;
30835 COLOR 14,1: LOCATE 20,47: PRINT USING "#####"; D25;
30840 COLOR 14,1: LOCATE 20,53: PRINT USING "##"; D26;
30845 COLOR 14,1: LOCATE 21,47: PRINT USING "#####"; D27;
30850 COLOR 14,1: LOCATE 21,53: PRINT USING "##"; D28;
30855 COLOR 14,1: LOCATE 22,47: PRINT USING "#####"; D29;
30860 COLOR 14,1: LOCATE 22,53: PRINT USING "##"; D30;
30865 COLOR 14,1: LOCATE 23,47: PRINT USING "#####"; D31;
30870 COLOR 14,1: LOCATE 23,53: PRINT USING "##"; D32;
30875 COLOR 14,1: LOCATE 6,69: PRINT USING "#########"; E1;
30880 COLOR 14,1: LOCATE 8,58: PRINT USING "########"; E2;
30885 COLOR 14,1: LOCATE 8,69: PRINT USING "#########"; E3;
30890 COLOR 14,1: LOCATE 10,58: PRINT USING "########"; E4;
30895 COLOR 14,1: LOCATE 10,69: PRINT USING "#########";
E5;
30900 COLOR 14,1: LOCATE 12,58: PRINT USING "########"; E6;
30905 COLOR 14,1: LOCATE 12,69: PRINT USING "#########";
E7;
30910 COLOR 14,1: LOCATE 14,58: PRINT USING "########"; E8;
30915 COLOR 14,1: LOCATE 14,69: PRINT USING "#########";
E9;
30920 COLOR 14,1: LOCATE 16,58: PRINT USING "########";
E10;
30925 COLOR 14,1: LOCATE 16,69: PRINT USING "#########";
E11;
30930 COLOR 14,1: LOCATE 18,58: PRINT USING "########";
E12;
30935 COLOR 14,1: LOCATE 18,69: PRINT USING "#########";
E13;
30940 COLOR 14,1: LOCATE 20,58: PRINT USING "########";
E14;
30945 COLOR 14,1: LOCATE 20,69: PRINT USING "#########";
E15;
30950 COLOR 14,1: LOCATE 22,58: PRINT USING "########";
E16;
30955 COLOR 14,1: LOCATE 22,69: PRINT USING "#########";
E17;
30960 COLOR 15,1: LOCATE 24,36: PRINT USING "######.#";
TOTALDIST;
30965 COLOR 15,1: LOCATE 24,47: PRINT USING "#####"; F1;
30970 COLOR 15,1: LOCATE 24,53: PRINT USING "##"; F2;
30975 COLOR 15,1: LOCATE 24,58: PRINT USING "########"; F3;
30980 COLOR 15,1: LOCATE 24,69: PRINT USING "#########";
F4;
30985 RETURN
30990 '
30995 '    Screen Display Initialization Statements
31000 '
31005 NUMFLDS.SS%=0: FILNM.SS$="LLJETLOG.SCR":
31010 EXITCHR.SS$=CHR$(27)+CHR$(127)+""
```

```
31015 RESTORE 30470
31020 RETURN
60000 '
60085 IF SAME.SS% THEN 60200  'To return to the SAME screen
with SAME values
60090 IF SCR.SS%=SCRLST.SS% THEN 60140  ' If same screen as
last, don't reload
60100                                   ' Get screen setup
parameters
60105    ON SCR.SS% GOSUB   30225, 30375, 30995
60110    GOSUB 60550                    ' Read field data
for this screen
60120    OUT &H3D8,&H1                  ' Turn off screen
display
60125 '  CALL QUBLOAD(FILNM.SS$)        '<- For QuickBASIC,
See READ.ME file
60130    DEF SEG=SCRNSEG.SS% : BLOAD FILNM.SS$,0 :DEF SEG '
Load screen picture
60135                   ' Set initial values
60140    IF INIT.SS% THEN ON SCR.SS% GOSUB   30095, 30310,
30450
60145                   ' Assign current values to screen
array
60155    ON SCR.SS% GOSUB   30005, 30260, 30410
60160    OUT &H3D8,&H29                 ' Turn on screen
display
60170    GOSUB 60590                    ' Pad fields with
blanks and display
60175                   ' Display initial DISPLAY variables
60180    ON SCR.SS% GOSUB   30130, 30335, 30470
60185    OUT &H3D8,&H29                 ' Turn on screen
display
60190 '
60195 F.SS%=1 : SCRLST.SS%=SCR.SS%
60200 COLOR 7,0:LOCATE 25,1:PRINT BLNK.SS$;   'Clr msg from
prior screen
60205 IF NUMFLDS.SS%=0 THEN RETURN            'Exit if no
fields on screen
60210 LOCATE ,,,0,13                          'Make cursor
size large
60215 EXSCR.SS%=0                             'Initialize
Flag For Screen Exit
60220 WHILE NOT EXSCR.SS%        'Loop on each field(until
Exit Flag is set)
60250     GOSUB 60740                         'Accept input
data for this field
60260     '
60320 WEND
60360 '
60370 '
```

```
60390 COLOR 7,0:LOCATE 25,1:PRINT BLNK.SS$;:LOCATE
25,15:PRINT "... Please WAIT A Moment While Checking Fields
...";
60400 FOR F.SS%=1 TO NUMFLDS.SS%      ' Test Each Field
Before Leaving Screen
60410     GOSUB 62500                 ' Check contents of
this field
60420     IF ERR.MSG%=-1 THEN EXSCR.SS%=0 : GOTO 60220
'Error Detected
60430 NEXT F.SS%
60440 F.SS%=FLDLST.SS%                          'Reset Field
indicator
60450 '
60455                                           'Assign new
field values
60460 ON SCR.SS% GOSUB   30050, 30285, 30430
60470 RETURN                                    'Exit this
subroutine
60480 '
60490
'*************************************************************
**************
60500 '
60510 '
60520 ' ****************************************
60530 ' **** Read Field Data For This Screen ****
60540 ' ****************************************
60550 FOR F.SS%=1 TO NUMFLDS.SS%
60555   READ
LO.SS%(F.SS%,2),LO.SS%(F.SS%,1),LE.SS%(F.SS%),TY.SS$(F.SS%)
,PIC.SS$(F.SS%),RG.SS(F.SS%,1),RG.SS(F.SS%,2),CL.SS%(F.SS%,
1),CL.SS%(F.SS%,2),SPECCHR.SS%(F.SS%)
60560 NEXT F.SS%
60565 RETURN
60570 '
60575
'*************************************************************
**************
60580 '*** Pad Fields With Blanks, Insert Special
Characters, and Display Flds
60585
'*************************************************************
**************
60590 FOR F.SS%=1 TO NUMFLDS.SS%
60595     IF TY.SS$(F.SS%)="N" THEN 60655   ' This section
for non-numeric types
60600        IF LEN(VL.SS$(F.SS%))>LE.SS%(F.SS%) THEN
VL.SS$(F.SS%)=LEFT$(VL.SS$(F.SS%),LE.SS%(F.SS%)): GOTO
60610
```

```
60605
VL.SS$(F.SS%)=VL.SS$(F.SS%)+MID$(BLNK.SS$,1,LE.SS%(F.SS%)-
LEN(VL.SS$(F.SS%)))
60610         IF INSTR("CD",TY.SS$(F.SS%))=0 OR
SPECCHR.SS%(F.SS%)=0 THEN 60690
60615           CNT.SS%=0
60620           FOR J.SS%=1 TO LE.SS%(F.SS%)          '
Insert Special chars
60625               IF
INSTR("ULX#89",MID$(PIC.SS$(F.SS%),J.SS%,1))=0 THEN
MID$(VL.SS$(F.SS%),J.SS%,1)=MID$(PIC.SS$(F.SS%),J.SS%,1) :
CNT.SS%=CNT.SS%+1
60630               IF CNT.SS%=SPECCHR.SS%(F.SS%) THEN
60690
60635           NEXT J.SS%
60640         GOTO 60690          ' End of "non-numeric
type" section
60645 '
60650 '                           ' The following section is
for numeric types
60655     NUMDEC%=LE.SS%(F.SS%)-INSTR(PIC.SS$(F.SS%),".")
' Calc # of dec places
60657     IF LEFT$(VL.SS$(F.SS%),1)=" " THEN
VL.SS$(F.SS%)=RIGHT$(VL.SS$(F.SS%),LEN(VL.SS$(F.SS%))-1) '
Strip leading blank
60660     IF NUMDEC%=LE.SS%(F.SS%) THEN NUMDEC%=0:
NUMINT%=LE.SS%(F.SS%) ELSE NUMINT%=LE.SS%(F.SS%)-NUMDEC%-1
' Calc # of interger places
60665     IF VAL(VL.SS$(F.SS%))=0 THEN
VL.SS$(F.SS%)=LEFT$(MID$(BLNK.SS$,1,NUMINT%-
1)+"0."+STRING$(NUMDEC%,"0"),LE.SS%(F.SS%)): GOTO 60690
' If no initial value
60670         DEC.VL%=INSTR(VL.SS$(F.SS%),".") : IF
DEC.VL%=0 THEN DEC.VL%=LE.SS%(F.SS%)+1
' Position of decimal point in data
60675
VL.SS$(F.SS%)=LEFT$(RIGHT$(MID$(BLNK.SS$,1,NUMINT%)+LEFT$(V
L.SS$(F.SS%),DEC.VL%-
1),NUMINT%)+"."+MID$(VL.SS$(F.SS%),DEC.VL%+1)+STRING$(NUMDE
C%,"0"),LE.SS%(F.SS%))
60680 '
60685 '
60690 GOSUB 62310                 ' Display Contents Of This
Field
60695 NEXT F.SS%
60700 RETURN
60705 '
60710 '
60715 '
*********************************************
```

```
60720 '          *** Accept Input Data For The Current
Field ***
60725 '
*******************************************
60730 '
60740 IF TY.SS$(F.SS%)<>"N" THEN A.SS%=1 : GOTO  60790
60750     NEWNUM%=-1:NUMED.SS%=0      'Set Flags for num
fld
60760     DECPOS%=INSTR(PIC.SS$(F.SS%),".") : IF DECPOS%=0
THEN DECPOS%=LE.SS%(F.SS%)+1
60770     A.SS%=DECPOS%-1
60780 '                                   ' Look for non-
edit characters
60790 WHILE INSTR("ULX#98",MID$(PIC.SS$(F.SS%),A.SS%,1))=0:
A.SS%=A.SS%+1: WEND
60800 CURCOL%=LO.SS%(F.SS%,2)+A.SS%-1     ' Find cursor
position on screen
60810 LOCATE LO.SS%(F.SS%,1),CURCOL%,1    ' Starting
position in this field
60820 '
60830 FLDLST.SS% = F.SS%                  ' Reset field
indicator
60840 EXFLD.SS%=0                         ' Initializei flag
to exit this field
60850 WHILE NOT EXFLD.SS%                 ' Loop while still
editing this field
60860   X.SS$=INKEY$ : IF X.SS$="" THEN 60860  ' Wait for
next keyboard character
60870   IF ERR.MSG% THEN ERR.MSG%=0 : COLOR 7,0:LOCATE
25,1,0:PRINT STRING$(79," ");
' Erase old message line.
60880   IF LEN(X.SS$)>1 OR
INSTR(CHR$(8)+CHR$(13)+CHR$(27),X.SS$)<>0 THEN
X.SS$=RIGHT$(X.SS$,1) ELSE 60960                    '
Extented code key pressed?
60885      ON INSTR(";<=>?@ABCD",X.SS$) GOSUB
61100,61100,61100,61100,61100,61100,61100,61100,61100,61100
' This is a DUMMY statement.  It traps the Function Keys
(F1 - F10).  It is here to make user modifications simpler.
60887              ' For the above line to be active, you
need to set all Function Key values to null. i.e.-  KEY
1,"" ; KEY 2,"" etc.
60890      IF TY.SS$(F.SS%)<>"N" THEN 60920       'If
numeric, need special test
60900         IF INSTR("GO",X.SS$)<>0 THEN 61030  'Codes
not valid for numeric
60910         IF INSTR("RKM"+CHR$(8),X.SS$)<>0 THEN
NUMED.SS%=-1:NEWNUM%=0
60920      IF INSTR(EXITCHR.SS$,X.SS$)<>0 THEN EXFLD.SS%=-
1:EXSCR.SS%=-1:GOTO 61040  'Check CODE to EXIT SCREEN
```

```
60930       ON INSTR("MKHPGRSO"+CHR$(8)+CHR$(13),X.SS$)
GOSUB
61110,61140,61210,61260,61650,61440,61300,61600,61140,61260
: GOTO 61020
60940       GOTO 61030              ' Invalid extended code
key pressed
60950 '
60960    IF TY.SS$(F.SS%)="N" AND X.SS$="." THEN
NEWNUM%=0:NUMED.SS%=-1: GOTO 61010
60970    IF ASC(X.SS$)<32 OR ASC(X.SS$)>126 THEN 61030    '
Invalid characters
60980    GOSUB 61750 :IF ERR.MSG% THEN 61030 ' Non-spec char
entered: Test entry
60990    GOSUB 62230                              ' Add char to
this field
61000    GOSUB 62310                              ' Print new
field
61010    GOSUB 62380                              ' Move cursor
to next position
61020    IF ERR.MSG% THEN  60740                  ' If error re-
edit this field
61030 WEND
61040 LASTCHR.SS$=X.SS$                             ' Set last
character indicator
61050 RETURN                                      ' Exit from
editing this field
61060 '
61070
'************************************************************
*************
61080 '****  THE FOLLOWING SUBROUTINES HANDLE EXTENDED
KEYBOARD COMMANDS  ****
61090
'************************************************************
*************
61095 ' ***  DUMMY Subroutine for Function Keys (F1 - F10)
61096 ' ***  This is here for user modifications   ***
61100 RETURN
61105 '*** Cursor right ***
61110    GOSUB 62380             ' Move cursor to next
position
61120    RETURN
61130 '*** Cursor left or backspace ***
61140    IF CURCOL%=LO.SS%(F.SS%,2) THEN 61210       'Goto
prior field
61150    IF TY.SS$(F.SS%)="N" AND INSTR(" +-
",MID$(VL.SS$(F.SS%),A.SS%,1))<>0 THEN RETURN
61160    CURCOL%=CURCOL%-1:A.SS%=A.SS%-1
'Pos of char in field
61170    IF INSTR("ULX#89",MID$(PIC.SS$(F.SS%),A.SS%,1))=0
THEN 61140 'Spec char?
```

```
61180    LOCATE LO.SS%(F.SS%,1),CURCOL%,1
61190    RETURN
61200 '*** Cursor up ***      Move to next field left
61210    GOSUB 62500                    ' Edit check field
data before leaving
61220    IF ERR.MSG% THEN RETURN        ' Error found
61225    EXFLD.SS%=-1                   ' Set Flag to Exit
this Field
61230    IF F.SS%>1 THEN F.SS%=F.SS%-1 ELSE
F.SS%=NUMFLDS.SS%                                 ' Goto
prior fld
61240    RETURN
61250 '*** Cursor down or carriage return - Advance to next
field ***
61260    GOSUB 62500                  ' Edit check field
data before leaving
61270    IF ERR.MSG% THEN  RETURN     ' Don't leave field
if error was found
61275    EXFLD.SS%=-1                 ' Set Flag to Exit
this Field
61277    IF F.SS%=NUMFLDS.SS% AND
INSTR(EXITCHR.SS$,CHR$(127))<>0 THEN EXSCR.SS%=-1  'Test to
leave screen after last field
61280    IF F.SS%<NUMFLDS.SS% THEN F.SS%=F.SS%+1 ELSE
F.SS%=1                           ' Increment fld num to
next fld
61290    RETURN
61300 ' **** Del key pressed ******
61310 '              ' Start Del routine for Numeric fld on
left of decimal pt
61320 IF TY.SS$(F.SS%)="N" AND A.SS%<DECPOS% THEN
MID$(VL.SS$(F.SS%),1)=" "+LEFT$(VL.SS$(F.SS%),A.SS%-
1)+RIGHT$(VL.SS$(F.SS%),LE.SS%(F.SS%)-A.SS%): GOTO 61420
61330 '              ' Start Del routine for Numeric fld on
right of decimal pt
61340 IF TY.SS$(F.SS%)="N" THEN
MID$(VL.SS$(F.SS%),1)=LEFT$(VL.SS$(F.SS%),A.SS%-
1)+MID$(VL.SS$(F.SS%),A.SS%+1,LE.SS%(F.SS%)-A.SS%)+"0" :
GOTO 61420
61350 '              ' Start Del routine for fld w/o non-
edit chr
61360 IF SPECCHR.SS%(F.SS%)=0 THEN
MID$(VL.SS$(F.SS%),1)=LEFT$(VL.SS$(F.SS%),A.SS%-
1)+MID$(VL.SS$(F.SS%),A.SS%+1,LE.SS%(F.SS%)-A.SS%)+" " :
GOTO 61420
61370 '
61380 CNT.SS%=0                  ' Start del routine for
fld with non-edit chr
61390 WHILE
INSTR("ULX#89",MID$(PIC.SS$(F.SS%),A.SS%+1+CNT.SS%,1))<>0
```

```
AND CNT.SS%<LE.SS%(F.SS%)-A.SS% : CNT.SS%=CNT.SS%+1 :  WEND
' Count until next non-edit chr
61400 VL.SS$(F.SS%)=LEFT$(VL.SS$(F.SS%),A.SS%-
1)+MID$(VL.SS$(F.SS%),A.SS%+1,CNT.SS%)+"
"+RIGHT$(VL.SS$(F.SS%),LE.SS%(F.SS%)-A.SS%-CNT.SS%)     '
New value for field
61410 '
61420 CURCOL%=CURCOL%-1:A.SS%=A.SS%-1:GOSUB 62340:GOSUB
62380 'Print fld; Set cursor
61430 RETURN
61440 ' ***** Ins key pressed ***
61450 '            ' Start Ins routine for numeric field on
leftside of dec pt
61460 IF TY.SS$(F.SS%)="N" AND A.SS%<DECPOS% THEN
MID$(VL.SS$(F.SS%),1)=MID$(VL.SS$(F.SS%),2,A.SS%-
1)+"0"+RIGHT$(VL.SS$(F.SS%),LE.SS%(F.SS%)-A.SS%) : GOTO
61580
61470 '            ' Start Ins routine for numeric field on
rightside of dec pt
61480 IF TY.SS$(F.SS%)="N"  THEN
VL.SS$(F.SS%)=LEFT$(VL.SS$(F.SS%),A.SS%-
1)+"0"+MID$(VL.SS$(F.SS%),A.SS%,LE.SS%(F.SS%)-A.SS%) : GOTO
61580
61490 '            ' Start Ins routine for non-numeric w/o
non-edit characters
61500 IF SPECCHR.SS%(F.SS%)=0 THEN
VL.SS$(F.SS%)=LEFT$(VL.SS$(F.SS%),A.SS%-1)+"
"+MID$(VL.SS$(F.SS%),A.SS%,LE.SS%(F.SS%)-A.SS%) : GOTO
61580
61510 '            ' Start Ins routine for non-numeric with
non-edit characters
61520 NEWVL$=LEFT$(VL.SS$(F.SS%),A.SS%-1)+" " :
NEXTCHR$=MID$(VL.SS$(F.SS%),A.SS%,1)
61530 FOR I%=A.SS%+1 TO LE.SS%(F.SS%)
61540   X.SS$=MID$(PIC.SS$(F.SS%),I%,1): IF
INSTR("ULX#89",X.SS$)=0 THEN NEWVL$=NEWVL$+X.SS$ : GOTO
61570
61550   NEWVL$=NEWVL$+NEXTCHR$ :
NEXTCHR$=MID$(VL.SS$(F.SS%),I%,1)
61560 NEXT I%
61570 .
VL.SS$(F.SS%)=NEWVL$+MID$(VL.SS$(F.SS%),I%+1,LE.SS%(F.SS%))
61580 CURCOL%=CURCOL%-1:A.SS%=A.SS%-1:GOSUB 62310:GOSUB
62380 ' Print fld; Set cursor
61590 RETURN
61600 ' ***** END key pressed *****
61610 CURCOL%=LO.SS%(F.SS%,2)+LE.SS%(F.SS%)-1  :
A.SS%=LE.SS%(F.SS%)
61620 WHILE INSTR("ULX#89",MID$(PIC.SS$(F.SS%),A.SS%,1))=0
:A.SS%=A.SS%-1:CURCOL%=CURCOL%-1: WEND       ' Look for
special protected characters
```

```
61630 LOCATE LO.SS%(F.SS%,1),CURCOL%,1       ' Starting
position in this field
61640 RETURN
61650 ' **** HOME key pressed ****  put cursor at beginning
of field
61660 A.SS%=1                        'Find cursor position for
this field
61670 WHILE INSTR("ULX#89",MID$(PIC.SS$(F.SS%),A.SS%,1))=0
: A.SS%=A.SS%+1 : WEND     ' Look for special protected
characters
61680 CURCOL%=LO.SS%(F.SS%,2)+A.SS%-1
61690 LOCATE LO.SS%(F.SS%,1),CURCOL%,1 ' Starting position
in this field
61700 RETURN
61710 '
61715 '
***********************************************
61720 '        ****  ROUTINE TO EDIT-TEST CHARACTER ENTRY
****
61730 '
***********************************************
61740 '***Check for special character type conversion***
61750 ON INSTR("NDMY",TY.SS$(F.SS%)) GOTO
61790,62100,62000,61950
61760 ON INSTR("ULX#89",MID$(PIC.SS$(F.SS%),A.SS%,1)) GOTO
61890,61920,62130,61800,62060,62100
61770 PRINT "EDIT PICTURE TYPE
";MID$(PIC.SS$(F.SS%),A.SS%,1);" NOT FOUND" : STOP
61780 '
61790 '*** Numeric values; "."; "-"; "+"; " "
61800 ' NOTE: The decimal point is trapped in the"Accept
input data" routine
61810 IF X.SS$>="0" AND X.SS$<="9" THEN RETURN
61820 IF X.SS$<>"+" AND X.SS$<>"-" AND X.SS$<>" " THEN
61850  ' Is this a + or - sign?
61830 IF TY.SS$(F.SS%)="C" THEN RETURN ' +,-," " allowed
anywhere in C type
61840    IF LEFT$(VL.SS$(F.SS%),A.SS%-1)=SPACE$(A.SS%-1) OR
A.SS%=1 THEN RETURN
61841 '      " "  "+" and "-" sign only allowed in
beginning of number
61850    MSG.SS$=" Only numeric values can be entered here.
Please re-enter. "
61860    GOSUB 62180 : RETURN        ' Print error message;
Exit
61870 '
61880 '*** Upper case or any other character***
61890 IF ASC(X.SS$)>96 AND ASC(X.SS$)<123 THEN
X.SS$=CHR$(ASC(X.SS$)-32)
61900 GOTO 62130
61910 '*** Lower case or any other character***
```

```
61920 IF ASC(X.SS$)>65 AND ASC(X.SS$)<91 THEN
X.SS$=CHR$(ASC(X.SS$)+32)
61930 GOTO 62130
61940 '*** Y/N answer only***
61950 IF X.SS$="Y" OR X.SS$="y" THEN X.SS$="Y" : GOTO 62130
61960 IF X.SS$="N" OR X.SS$="n" THEN X.SS$="N" : GOTO 62130
61970    MSG.SS$=" Only 'Y' or 'N' can be entered here.
Please re-enter. "
61980    GOTO 62180          ' Print error message
61990 '*** M/F answer only***
62000 IF X.SS$="M" OR X.SS$="m" THEN X.SS$="M" : GOTO 62130
62010 IF X.SS$="F" OR X.SS$="f" THEN X.SS$="F" : GOTO 62130
62020    MSG.SS$=" Only 'M' or 'F' can be entered here.
Please re-enter. "
62030    GOTO 62180        ' Print error message
62040 '
62050 ' *** Numeric values only ****
62060 IF (ASC(X.SS$)>47 AND ASC(X.SS$)<58) THEN GOTO 62130
62070    MSG.SS$=" Only numeric values can be entered here.
Please re-enter. "
62080    GOTO 62180          ' Print error message
62090 ' *** Numeric values and " " only ***
62100 IF (ASC(X.SS$)>47 AND ASC(X.SS$)<58) OR X.SS$=" "
THEN GOTO 62130
62110    MSG.SS$=" Only numeric values or blanks can be
entered here. Please re-enter. "
62120    GOTO 62180          ' Print error message
62130 RETURN
62140 '
62150 ' ********************************
62160 ' **** Print Error Messages ******
62170 ' ********************************
62180   ERR.MSG%=-1 : SOUND 500,SD.SS%*1:LOCATE 25,INT(81-
LEN(MSG.SS$))/2,0 :COLOR 0,7:PRINT MSG.SS$;: LOCATE
LO.SS%(F.SS%,1),CURCOL%,1
62190   WHILE INKEY$<>"" : WEND     ' Clear input buffer
after error
62200 RETURN
62210 '
62220 ' ****************************************
62230 ' **** Add character to current field *****
62240 ' ****************************************
62250 IF TY.SS$(F.SS%)="N" AND NEWNUM% THEN
MID$(VL.SS$(F.SS%),1)=MID$(BLNK.SS$,1,A.SS%-
1)+X.SS$+"."+STRING$(LE.SS%(F.SS%),"0") : NEWNUM%=0 :
RETURN  ' Restart fld - New num
62260 IF TY.SS$(F.SS%)<>"N" OR NUMED.SS%=-1 THEN
MID$(VL.SS$(F.SS%),A.SS%,1)=X.SS$ : RETURN
62270 IF LEFT$(VL.SS$(F.SS%),1)=" " THEN
MID$(VL.SS$(F.SS%),1,A.SS%)=MID$(VL.SS$(F.SS%),2,A.SS%-
```

```
1)+X.SS$  ELSE  NUMED.SS%=-1 : GOTO 62260    ' Add
character to left of decimal place
62280 RETURN
62290 '
62300 '*******************************
62310 '*** Print new value of field ***
62320 '*******************************
62330 '
62340 COLOR CL.SS%(F.SS%,1),CL.SS%(F.SS%,2)          ' Set
color for this field
62350 LOCATE LO.SS%(F.SS%,1),LO.SS%(F.SS%,2),0 : PRINT
VL.SS$(F.SS%); ' Print field
62360 RETURN
62365 '
62370 '************************************
62380 '**** Move cursor to new location ****
62390 '************************************
62400 IF TY.SS$(F.SS%)<>"N" OR NUMED.SS%<>0 THEN 62420
62410    IF LEFT$(VL.SS$(F.SS%),1)<>" " THEN NUMED.SS%=-1
ELSE A.SS%=DECPOS%-1:CURCOL%=LO.SS%(F.SS%,2)+A.SS%-1:LOCATE
LO.SS%(F.SS%,1),CURCOL%,1 : RETURN
62415 '
62420 IF A.SS%<LE.SS%(F.SS%) THEN
A.SS%=A.SS%+1:CURCOL%=CURCOL%+1 ELSE GOSUB 61260 : RETURN
' Advance cursor or go to next field
62430     IF INSTR("ULX#89",MID$(PIC.SS$(F.SS%),A.SS%,1))=0
THEN 62420
62440     LOCATE LO.SS%(F.SS%,1),CURCOL%,1
62450 RETURN
62460 '
62470 ' ************************************
62480 ' ***  Edit check final field result ***
62490 ' ************************************
62500 IF TY.SS$(F.SS%)<>"N" THEN 62540    ' Check numeric
input range
62510     IF VAL(VL.SS$(F.SS%))>RG.SS(F.SS%,2) THEN
MSG.SS$=" The maximum value allowed in this field is
"+STR$(RG.SS(F.SS%,2)): GOSUB 62180 : RETURN  'Print Err
Msg
62520     IF VAL(VL.SS$(F.SS%))<RG.SS(F.SS%,1) THEN
MSG.SS$=" The minimum value allowed in this field is
"+STR$(RG.SS(F.SS%,1)): GOSUB 62180 : RETURN  'Print Err
Msg
62530 '
62540 IF TY.SS$(F.SS%)<>"D" THEN 62720         ' Date edit
check
62550     IF VL.SS$(F.SS%)="  /  /  " THEN 62720
' No check
62560
DT.SS=VAL(RIGHT$(VL.SS$(F.SS%),2)+LEFT$(VL.SS$(F.SS%),2)+MI
D$(VL.SS$(F.SS%),4,2))
```

```
62570     IF
INSTR("01,02,03,04,05,06,07,08,09,10,11,12",LEFT$(VL.SS$(F.
SS%),2))<>0 THEN 62590
62580         MSG.SS$="Invalid MONTH in date entered. Please
re-enter.":GOSUB 62180 : RETURN
62590     IF INSTR(MID$(VL.SS$(F.SS%),4,2)," ")=0 AND
VAL(MID$(VL.SS$(F.SS%),4,2))>0 AND
VAL(MID$(VL.SS$(F.SS%),4,2))<32 THEN 62620
62600         MSG.SS$="Invalid DAY in date entered.  Please
re-enter.":GOSUB 62180:RETURN
62610 '
62620     IF RG.SS(F.SS%,1)=0 AND RG.SS(F.SS%,2)=0 THEN
62720  ' No check
62630     IF RG.SS(F.SS%,2)<RG.SS(F.SS%,1) THEN 62680
'Does date cross century?
62640         IF DT.SS>=RG.SS(F.SS%,1) AND
DT.SS<=RG.SS(F.SS%,2) THEN 62720
62650             D1.SS$=STR$(RG.SS(F.SS%,1)) :
D2.SS$=STR$(RG.SS(F.SS%,2))
62660             MSG.SS$="The date should be between
"+MID$(D1.SS$,4,2)+"/"+RIGHT$(D1.SS$,2)+"/"+MID$(D1.SS$,2,2
)+" and
"+MID$(D2.SS$,4,2)+"/"+RIGHT$(D2.SS$,2)+"/"+MID$(D2.SS$,2,2
) :GOSUB 62180 : RETURN
62670 '
62680     IF DT.SS>=RG.SS(F.SS%,2) OR DT.SS<=RG.SS(F.SS%,1)
THEN 62720
62690             D1.SS$=STR$(RG.SS(F.SS%,2)) :
D2.SS$=STR$(RG.SS(F.SS%,1))
62700             MSG.SS$="The date should be between
"+MID$(D1.SS$,4,2)+"/"+RIGHT$(D1.SS$,2)+"/"+MID$(D1.SS$,2,2
)+" and
"+MID$(D2.SS$,4,2)+"/"+RIGHT$(D2.SS$,2)+"/"+MID$(D2.SS$,2,2
) :GOSUB 62180 : RETURN
62710 '
62720 RETURN
62860 ' ************************************************
62870 ' ** Test If Monochrome or Color/Graphics Monitor **
62880 ' ************************************************
62890 DEF SEG=&H40
62900 MONO.SS=(PEEK(&H10) AND &H30)=&H30
62910 IF MONO.SS THEN SCRNSEG.SS%=&HB000 ELSE
SCRNSEG.SS%=&HB800
62920 RETURN
62930 ' ******************************
62940 ' **  Error Handling Routine  **
62950 ' ******************************
62960 ' This routine is used mostly to turn the screen
display back on if
62970 ' a serious problem (such as the screen image file
not found) occurs.
```

```
62980 CLS:OUT &H3D8,&H29          ' Turn screen display BACK
ON (if off).
62990                        'The following checks for disk
file errors
63000 IF NOT (ERR=53 OR ERR=57 OR ERR=66 OR (ERR>70 AND
ERR<77)) THEN 63030
63010    LOCATE 10,1:PRINT"There is a problem finding a
file on the disk.   Error code="+STR$(ERR)
63020    LOCATE 11,1:PRINT"(HINT:Check disk for .SCR
file)":PRINT
63030 ON ERROR GOTO 0          ' Reset BASIC ERROR handling
```

```
0 '    MAXRANGE - 7/27/1986 - 11:25:35
2 COPYRIGHT.SS$="(C)Copyright 84,85 The Software Bottling
Company Of New York"
5 '
10 DIM VL.SS$(25), LO.SS%(25,2), LE.SS%(25), TY.SS$(25),
PIC.SS$(25),RG.SS(25,2), CL.SS%(25,2), SPECCHR.SS%(25)
20 KEY OFF: SD.SS%=1 :. NUMSCR.SS%=2 : BLNK.SS$=SPACE$(78)
30 GOSUB 62890 'Test For Mono OR Color Display
100 '********** OPTIMUM PERFORMANCE PROGRAM **********
110 CLS:A=0:B=0:C=0:PRINT "~L=MAX/"
111 LOCATE 1,1,0:PRINT "~W=MAXINTRO/":SCREEN ,,3,0:INPUT
"",HARD:SCREEN ,,0,0
120 SCR.SS%=1:INIT.SS%=-1:GOSUB 60000           'input
screen display
130 DC=DC6:SELECTALT2=SELECTALT
135 SOXDC=DC
140 IF DC<=100 THEN GOSUB 500 ELSE GOSUB 600
150 MACHNUM=BRMN:GOSUB 400
160 IF (ABS(SOXDC-DC))>5 THEN SOXDC=DC:GOTO 140
170 DCOPTALT=SOXDC:MRMACH1=BRMN
180 DC=DC6
185 SOXDC=DC:ALT=SELECTALT
190 GOSUB 800:MACHNUM=BRMN:GOSUB 400
200 IF (ABS(SOXDC-DC))>5 THEN SOXDC=DC:GOTO 190
210 DCANYALT=SOXDC:MRMACH2=BRMN
220 DC=DCOPTALT:GOSUB 700:OPTALT1=ALT
230 DC=DCOPTALT:GOSUB 900:OPTSR1=BRSR
240 DC=DCANYALT:ALT=SELECTALT:GOSUB 900:OPTSR2=BRSR
250 DC=DCOPTALT:GOSUB 1000:OPTENDALT1=BEALT:ALT=BEALT:GOSUB
1100:OPTENDMACH1=BEMN:GOSUB 1200:OPTENDFF=BEFF
300 'COMPUTE TEMPERATURE AT OPTALT1 AND SELECTALT
310 DELTAT=(TEMP+459.69)-518.69
320 IF OPTALT1<=36089! THEN T1=518.69-
(.0035662*OPTALT1)+DELTAT ELSE T1=389.99+DELTAT
330 IF SELECTALT<=36089! THEN T2=518.69-
(.0035662*OPTALT)+DELTAT ELSE T2=389.99+DELTAT
340 A1=(49.01*SQR(T1))*.5921:A2=(49.01*SQR(T2))*.5921
'speed of sound kts
350 IF TAILWIND>0 THEN HEADWIND=TAILWIND*(-1)
355 TAS1=MRMACH1*A1:TAS2=MRMACH2*A2                    'no
wind condition
360 BRSRNOHW=OPTSR1:TAS=TAS1:GOSUB 1300:OPTSR1=BRSRHW
370 BRSRNOHW=OPTSR2:TAS=TAS2:GOSUB 1300:OPTSR2=BRSRHW
380 BRMNNOHW=MRMACH1:GOSUB 1400:MRMACH1=BRMNHW
390 BRMNNOHW=MRMACH2:GOSUB 1400:MRMACH2=BRMNHW
392 TAS1=MRMACH1*A1:TAS2=MRMACH2*A2
'corrected for headwind
393 GS1=TAS1-HEADWIND:GS2=TAS2-HEADWIND
394 HOURS1=INT(DIST/GS1):MINUTES1=(DIST/GS1-
HOURS1)*60:FUEL1=DIST*OPTSR1
```

```
395 HOURS2=INT(DIST/GS2):MINUTES2=(DIST/GS2-
HOURS2)*60:FUEL2=DIST*OPTSR2
396 LOCATE 1,1:SCR.SS%=2:INIT.SS%=0:GOSUB 60000:GOSUB
2000:LOCATE 1,1:PRINT "~P=OPTRNG/"
397 LOCATE 24,27:PRINT "PRESS ANY KEY TO CONTINUE";
398 ANS$=INKEY$:IF ANS$="" THEN 398
399 GOTO 1440.
400 '********** SOX DRAG EQUATION **********
410 DC=DC6*(1+(60*((MACHNUM-.6)^3)))
420 RETURN
500 '***** BEST RANGE MACH NUMBER AT CRUISE CEILING FOR DC
<= 100 *****
510 BRMN=.85816-(.000503*DC)+(1.6933E-06*DC^2)
520 RETURN
600 '***** BEST RANGE MACH NUMBER AT CRUISE CEILING FOR DC
> 100 *****
610 BRMN=1.0595-(2.93067E-03*DC)+(6.44E-06*DC^2)-(6.0333E-
09*DC^3)
620 RETURN
700 '***** OPTIMUM CRUISE ALTITUDE OR CRUISE CEILING *****
710  ALT=EXP(11.0605-(5.53315E-08*GW*DC)-(1.25963E-
05*GW)+(4.57508E-14*GW*DC^3))
720 RETURN
800 '***** BEST RANGE MACH NUMBER AT ANY ALTITUDE *****
810 BRMN=.454334-(7.13908E-04*DC)+(8.92715E-07*DC^2)-
(6.21055E-15*ALT^3)+(6.15457E-10*ALT^2)-(8.426901E-
06*ALT)+(5.74695E-11*GW^2)+(3.6246E-10*ALT*GW)-(1.2802E-
08*DC*ALT)+(3.76629E-13*ALT^2*DC)-(6.61474E-15*ALT^2*GW)
820 RETURN
900 '***** BEST RANGE SPECIFIC RANGE (lb/nm) *****
910 BRSR=EXP(-2.15494+(3.89145E-05*ALT)-(2.33982E-03*DC)-
(1.56937E-10*GW^2)-(3.55219E-15*ALT^3)-(4.36678E-
10*ALT*GW)+(1.74403E-06*DC^2)-(1.85822E-08*ALT*DC))
915 BRSR=1/BRSR
920 RETURN
1000 '***** OPTIMUM ENDURANCE ALTITUDE *****
1010 BEALT=48690.9-(.591744*DC^2)-(3.25866E-
12*DC*GW^3)+(2.32902E-05*DC^2*GW)-(.534385*GW)+(4.762322E-
31*GW^7*DC)-(7.98225E-11*DC^4*GW)+(2.35685E-06*DC^4)
1020 RETURN
1100 '***** MAXIMUM ENDURANCE MACH NUMBER *****
1110 BEMN=.175855+(5.71392E-06*GW)-(3.08863E-
04*DC)+(2.82951E-15*ALT^3)+(3.57714E-10*ALT*GW)+(3.62212E-
07*DC^2)-(4.22656E-15*GW^2*ALT)
1120 RETURN
1200 '***** MAXIMUM ENDURANCE FUEL FLOW *****
1210 BEFF=1950.85+(8.38134E-07*GW^2)-
(.0586557*ALT)+(1.6863E-04*GW*DC)+(7.97748E-
07*ALT^2)+(1.60371E-11*GW^2*ALT)+(5.14017E-05*ALT*DC)-
(1.82085*DC)
1220 RETURN
```

```
1300 '***** BEST RANGE SPECIFIC RANGE WITH A HEADWIND OR
TAILWIND *****
1310 BRSRHW=BRSRNOHW*((TAS-HEADWIND)/TAS)
1320 RETURN
1400 '***** BEST RANGE MACH NUMBER WITH A HEADWIND OR
TAILWIND *****
1410 BRMNHW=BRMNNOHW+(.000475*HEADWIND)
1420 RETURN
1440 CLS:LOCATE 1,1:SYSTEM
1500 END
2000 '********** PRINT SCREEN ASSEMBLY LANGUAGE SUBROUTINE
**********
2010 A%(0)=&HCD55
2020 A%(1)=&H5D05
2030 A%(2)=&H90CB
2040 SUBRT = VARPTR(A%(0))
2060 IF HARD=2 THEN RETURN ELSE CALL SUBRT
2070 LPRINT CHR$(12)
2080 RETURN
10000 ON ERROR GOTO 62980      ' Error Handling Routine -
Delete
10010                          ' if it conflicts with your
own
30000 '
30005 '    Variables Section For B:MAXRNG1
30010 '
30015 VL.SS$(1)=STR$(GW): VL.SS$(2)=STR$(DC6):
VL.SS$(3)=STR$(SELECTALT):
30020 VL.SS$(4)=STR$(DIST): VL.SS$(5)=STR$(HEADWIND):
30025 VL.SS$(6)=STR$(TAILWIND): VL.SS$(7)=STR$(TEMP):
30030 RETURN
30035 '
30040 '    Assign VL.SS$ Array to the variables
30045 '
30050
GW=VAL(VL.SS$(1)):DC6=VAL(VL.SS$(2)):SELECTALT=VAL(VL.SS$(3
)):
30055
DIST=VAL(VL.SS$(4)):HEADWIND=VAL(VL.SS$(5)):TAILWIND=VAL(VL
.SS$(6)):
30060 TEMP=VAL(VL.SS$(7)):
30065 RETURN
30070 '
30075 '     Section To Initialize Variables To Initial
Values
30080 '
30085 GW=0: DC6=0: SELECTALT=0: DIST=0: HEADWIND=0:
TAILWIND=0: TEMP=0:
30090 RETURN
30095 '
```

```
30100 '     ****  List DATA statements & Print DISPLAY Only
Variables  ****
30105 'Lin,Col,Len,Picture,Low Range,High
Range,Foreground,Background,# of Edit
30110 '
30115 DATA 58,9,5,"N","#####",0,42000,15,1,0
30120 DATA 60,11,3,"N","###",0,400,15,1,0
30125 DATA 58,14,5,"N","#####",0,45000,15,1,0
30130 DATA 59,16,4,"N","####",0,9999,15,1,0
30135 DATA 60,18,3,"N","###",0,300,15,1,0
30140 DATA 60,20,3,"N","###",0,300,15,1,0
30145 DATA 60,22,3,"N","###",-20,140,15,1,0
30150 RETURN
30155 '
30160 '    Screen Display Initialization Statements
30165 '
30170 NUMFLDS.SS%=7: FILNM.SS$="MAXRNG1.SCR":
30175 EXITCHR.SS$=CHR$(27)+CHR$(127)+""
30180 RESTORE 30100
30185 RETURN
30190 '
30195 '    Variables Section For B:MAXRNG2
30200 '
30205 RETURN
30210 '
30215 '    Assign VL.SS$ Array to the variables
30220 '
30225 RETURN
30230 '
30235 '     Section To Initialize Variables To Initial
Values
30240 '
30245 RETURN
30250 '
30255 '     ****  List DATA statements & Print DISPLAY Only
Variables  ****
30260 'Lin,Col,Len,Picture,Low Range,High
Range,Foreground,Background,# of Edit
30265 '
30270 COLOR 15,1: LOCATE 6,32: PRINT USING "#####"; GW;
30275 COLOR 15,1: LOCATE 7,32: PRINT USING "#####"; DC6;
30280 COLOR 15,1: LOCATE 8,32: PRINT USING "#####";
SELECTALT;
30285 COLOR 15,1: LOCATE 9,32: PRINT USING "#####"; DIST;
30290 COLOR 15,1: LOCATE 6,70: PRINT USING "####"; TEMP;
30295 COLOR 15,1: LOCATE 7,70: PRINT USING "####";
HEADWIND;
30300 COLOR 15,1: LOCATE 13,32: PRINT USING "#####";
OPTALT1;
30305 COLOR 15,1: LOCATE 14,32: PRINT USING "##.##";
MRMACH1;
```

```
30310 COLOR 15,1: LOCATE 15,32: PRINT USING "###.#";
OPTSR1;
30315 COLOR 15,1: LOCATE 16,32: PRINT USING "##"; HOURS1;
30320 COLOR 15,1: LOCATE 16,35: PRINT USING "##"; MINUTES1;
30325 COLOR 15,1: LOCATE 17,32: PRINT USING "#####"; FUEL1;
30330 COLOR 15,1: LOCATE 18,10: PRINT USING "###"; TAS1;
30335 COLOR 15,1: LOCATE 18,19: PRINT USING "###"; GS1;
30340 COLOR 15,1: LOCATE 21,32: PRINT USING "#####";
OPTENDALT1;
30345 COLOR 15,1: LOCATE 22,32: PRINT USING "##.##";
OPTENDMACH1;
30350 COLOR 15,1: LOCATE 23,32: PRINT USING "#####";
OPTENDFF;
30355 COLOR 15,1: LOCATE 13,69: PRINT USING "#####";
SELECTALT2;
30360 COLOR 15,1: LOCATE 14,69: PRINT USING "##.##";
MRMACH2;
30365 COLOR 15,1: LOCATE 15,69: PRINT USING "###.#";
OPTSR2;
30370 COLOR 15,1: LOCATE 16,69: PRINT USING "##"; HOURS2;
30375 COLOR 15,1: LOCATE 16,72: PRINT USING "##"; MINUTES2;
30380 COLOR 15,1: LOCATE 17,69: PRINT USING "#####"; FUEL2;
30385 COLOR 15,1: LOCATE 18,47: PRINT USING "###"; TAS2;
30390 COLOR 15,1: LOCATE 18,56: PRINT USING "###"; GS2;
30395 RETURN
30400 '
30405 '    Screen Display Initialization Statements
30410 '
30415 NUMFLDS.SS%=0: FILNM.SS$="MAXRNG2.SCR":
30420 EXITCHR.SS$=CHR$(27)+CHR$(127)+""
30425 RESTORE 30255
30430 RETURN
30435 RETURN
30440 EXITCHR.SS$=CHR$(27)+CHR$(127)+""
30445 RESTORE 30270
30450 RETURN
30455 RETURN
30460 RESTORE 30280
30465 RETURN
60000 '
60010 '
èëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëë
£
60020 ' ¤   START OF SCREEN SCULPTOR FULL SCREEN EDITING
ROUTINE    ¤
60030 '
àëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëëë
¥
60050 '                Ö---------------------------¢
60060 '                °  Main body of subroutine  °
60070 '                â---------------------------ì
```

```
60080 '
60085 IF SAME.SS% THEN 60200  'To return to the SAME screen
with SAME values
60090 IF SCR.SS%=SCRLST.SS% THEN 60140  ' If same screen as
last, don't reload
60100                                   ' Get screen setup
parameters
60105    ON SCR.SS% GOSUB   30160, 30405
60110    GOSUB 60550                    ' Read field data
for this screen
60120    OUT &H3D8,&H1                  ' Turn off screen
display
60125 '  CALL QUBLOAD(FILNM.SS$)        '<- For QuickBASIC,
See READ.ME file
60130    DEF SEG=SCRNSEG.SS% : BLOAD FILNM.SS$,0 :DEF SEG '
Load screen picture
60135                   ' Set initial values
60140    IF INIT.SS% THEN ON SCR.SS% GOSUB   30075, 30235
60145                   ' Assign current values to screen
array
60155    ON SCR.SS% GOSUB   30005, 30195
60160    OUT &H3D8,&H29                 ' Turn on screen
display
60170    GOSUB 60590                    ' Pad fields with
blanks and display
60175                   ' Display initial DISPLAY variables
60180    ON SCR.SS% GOSUB   30100, 30255
60185    OUT &H3D8,&H29                 ' Turn on screen
display
60190 '
60195 F.SS%=1 : SCRLST.SS%=SCR.SS%
60200 COLOR 7,0:LOCATE 25,1:PRINT BLNK.SS$;   'Clr msg from
prior screen
60205 IF NUMFLDS.SS%=0 THEN RETURN            'Exit if no
fields on screen
60210 LOCATE ,,,0,13                          'Make cursor
size large
60215 EXSCR.SS%=0                             'Initialize
Flag For Screen Exit
60220 WHILE NOT EXSCR.SS%        'Loop on each field(until
Exit Flag is set)
60250     GOSUB 60740                         'Accept input
data for this field
60260     '
60265     ' The above subroutine accepts data for a single
field.  The                     program will return to
this spot after the cursor exits any
field on the input screen.
60270     '
60275     ' If you want to do any special input field
testing, this is
```

```
60280     ' a good place to do it.
60285     '
60290     ' The following variables are passed back for
your use:
60295     ' F.SS% is next field to be edited
60300     ' FLDLST.SS% is last field edited
60305     ' LASTCHR.SS$ is last keyboard character entered
60310     ' VL.SS$(n) contains the current value of field
n.
60315     '
60320 WEND
60360 '
60370 '
60390 COLOR 7,0:LOCATE 25,1:PRINT BLNK.SS$;:LOCATE
25,15:PRINT "... Please WAIT A Moment While Checking Fields
...";
60400 FOR F.SS%=1 TO NUMFLDS.SS%      ' Test Each Field
Before Leaving Screen
60410     GOSUB 62500                 ' Check contents of
this field
60420     IF ERR.MSG%=-1 THEN EXSCR.SS%=0 : GOTO 60220
'Error Detected
60430 NEXT F.SS%
60440 F.SS%=FLDLST.SS%                          'Reset Field
indicator
60450 '
60455                                           'Assign new
field values
60460 ON SCR.SS% GOSUB   30040, 30215
60470 RETURN                                    'Exit this
subroutine
60480 '
60490
'*************************************************************
**************
60500 '
60510 '
60520 ' ****************************************
60530 ' **** Read Field Data For This Screen ****
60540 ' ****************************************
60550 FOR F.SS%=1 TO NUMFLDS.SS%
60555   READ
LO.SS%(F.SS%,2),LO.SS%(F.SS%,1),LE.SS%(F.SS%),TY.SS$(F.SS%)
,PIC.SS$(F.SS%),RG.SS(F.SS%,1),RG.SS(F.SS%,2),CL.SS%(F.SS%,
1),CL.SS%(F.SS%,2),SPECCHR.SS%(F.SS%)
60560 NEXT F.SS%
60565 RETURN
60570 '
60575
'*************************************************************
**************
```

```
60580 '*** Pad Fields With Blanks, Insert Special
Characters, and Display Flds
60585
'*************************************************************
**************
60590 FOR F.SS%=1 TO NUMFLDS.SS%
60595     IF TY.SS$(F.SS%)="N" THEN 60655   ' This section
for non-numeric types
60600         IF LEN(VL.SS$(F.SS%))>LE.SS%(F.SS%) THEN
VL.SS$(F.SS%)=LEFT$(VL.SS$(F.SS%),LE.SS%(F.SS%)): GOTO
60610
60605
VL.SS$(F.SS%)=VL.SS$(F.SS%)+MID$(BLNK.SS$,1,LE.SS%(F.SS%)-
LEN(VL.SS$(F.SS%)))
60610         IF INSTR("CD",TY.SS$(F.SS%))=0 OR
SPECCHR.SS%(F.SS%)=0 THEN 60690
60615             CNT.SS%=0
60620             FOR J.SS%=1 TO LE.SS%(F.SS%)          '
Insert Special chars
60625                 IF
INSTR("ULX#89",MID$(PIC.SS$(F.SS%),J.SS%,1))=0 THEN
MID$(VL.SS$(F.SS%),J.SS%,1)=MID$(PIC.SS$(F.SS%),J.SS%,1) :
CNT.SS%=CNT.SS%+1
60630                 IF CNT.SS%=SPECCHR.SS%(F.SS%) THEN
60690
60635             NEXT J.SS%
60640         GOTO 60690          ' End of "non-numeric
type" section
60645 '
60650 '                           ' The following section is
for numeric types
60655     NUMDEC%=LE.SS%(F.SS%)-INSTR(PIC.SS$(F.SS%),".")
' Calc # of dec places
60657     IF LEFT$(VL.SS$(F.SS%),1)=" " THEN
VL.SS$(F.SS%)=RIGHT$(VL.SS$(F.SS%),LEN(VL.SS$(F.SS%))-1) '
Strip leading blank
60660     IF NUMDEC%=LE.SS%(F.SS%) THEN NUMDEC%=0:
NUMINT%=LE.SS%(F.SS%) ELSE NUMINT%=LE.SS%(F.SS%)-NUMDEC%-1
' Calc # of interger places
60665     IF VAL(VL.SS$(F.SS%))=0 THEN
VL.SS$(F.SS%)=LEFT$(MID$(BLNK.SS$,1,NUMINT%-
1)+"0."+STRING$(NUMDEC%,"0"),LE.SS%(F.SS%)): GOTO 60690
' If no initial value
60670         DEC.VL%=INSTR(VL.SS$(F.SS%),".") : IF
DEC.VL%=0 THEN DEC.VL%=LE.SS%(F.SS%)+1
' Position of decimal point in data
60675
VL.SS$(F.SS%)=LEFT$(RIGHT$(MID$(BLNK.SS$,1,NUMINT%)+LEFT$(V
L.SS$(F.SS%),DEC.VL%-
1),NUMINT%)+"."+MID$(VL.SS$(F.SS%),DEC.VL%+1)+STRING$(NUMDE
C%,"0"),LE.SS%(F.SS%))
```

```
60680 '
60685 '
60690 GOSUB 62310                ' Display Contents Of This
Field
60695 NEXT F.SS%
60700 RETURN
60705 '
60710 '
60715 '
*********************************************
60720 '          *** Accept Input Data For The Current
Field ***
60725 '
*********************************************
60730 '
60740 IF TY.SS$(F.SS%)<>"N" THEN A.SS%=1 : GOTO  60790
60750     NEWNUM%=-1:NUMED.SS%=0       'Set Flags for num
fld
60760     DECPOS%=INSTR(PIC.SS$(F.SS%),".") : IF DECPOS%=0
THEN DECPOS%=LE.SS%(F.SS%)+1
60770     A.SS%=DECPOS%-1
60780 '                                  ' Look for non-
edit characters
60790 WHILE INSTR("ULX#98",MID$(PIC.SS$(F.SS%),A.SS%,1))=0:
A.SS%=A.SS%+1: WEND
60800 CURCOL%=LO.SS%(F.SS%,2)+A.SS%-1     ' Find cursor
position on screen
60810 LOCATE LO.SS%(F.SS%,1),CURCOL%,1    ' Starting
position in this field
60820 '
60830 FLDLST.SS% = F.SS%                ' Reset field
indicator
60840 EXFLD.SS%=0                       ' Initializei flag
to exit this field
60850 WHILE NOT EXFLD.SS%               ' Loop while still
editing this field
60860   X.SS$=INKEY$ : IF X.SS$="" THEN 60860  ' Wait for
next keyboard character
60870   IF ERR.MSG% THEN ERR.MSG%=0 : COLOR 7,0:LOCATE
25,1,0:PRINT STRING$(79," ");
' Erase old message line.
60880   IF LEN(X.SS$)>1 OR
INSTR(CHR$(8)+CHR$(13)+CHR$(27),X.SS$)<>0 THEN
X.SS$=RIGHT$(X.SS$,1) ELSE 60960                    '
Extented code key pressed?
60885      ON INSTR(";<=>?@ABCD",X.SS$) GOSUB
61100,61100,61100,61100,61100,61100,61100,61100,61100,61100
' This is a DUMMY statement.  It traps the Function Keys
(F1 - F10).  It is here to make user modifications simpler.
```

```
60887              ' For the above line to be active, you
need to set all Function Key values to null. i.e.-  KEY
1,"" ; KEY 2,"" etc.
60890      IF TY.SS$(F.SS%)<>"N" THEN 60920       'If
numeric, need special test
60900         IF INSTR("GO",X.SS$)<>0 THEN 61030  'Codes
not valid for numeric
60910         IF INSTR("RKM"+CHR$(8),X.SS$)<>0 THEN
NUMED.SS%=-1:NEWNUM%=0
60920      IF INSTR(EXITCHR.SS$,X.SS$)<>0 THEN EXFLD.SS%=-
1:EXSCR.SS%=-1:GOTO 61040  'Check CODE to EXIT SCREEN
60930      ON INSTR("MKHPGRSO"+CHR$(8)+CHR$(13),X.SS$)
GOSUB
61110,61140,61210,61260,61650,61440,61300,61600,61140,61260
: GOTO 61020
60940      GOTO 61030             ' Invalid extended code
key pressed
60950 '
60960   IF TY.SS$(F.SS%)="N" AND X.SS$="." THEN
NEWNUM%=0:NUMED.SS%=-1: GOTO 61010
60970   IF ASC(X.SS$)<32 OR ASC(X.SS$)>126 THEN 61030   '
Invalid characters
60980   GOSUB 61750 :IF ERR.MSG% THEN 61030 ' Non-spec char
entered: Test entry
60990   GOSUB 62230                          ' Add char to
this field
61000   GOSUB 62310                          ' Print new
field
61010   GOSUB 62380                          ' Move cursor
to next position
61020   IF ERR.MSG% THEN  60740              ' If error re-
edit this field
61030 WEND
61040 LASTCHR.SS$=X.SS$                        ' Set last
character indicator
61050 RETURN                                 ' Exit from
editing this field
61060 '
61070
'************************************************************
*************
61080 '****  THE FOLLOWING SUBROUTINES HANDLE EXTENDED
KEYBOARD COMMANDS  ****
61090
'************************************************************
*************
61095 ' ***  DUMMY Subroutine for Function Keys (F1 - F10)
61096 ' ***  This is here for user modifications   *** .
61100 RETURN
61105 '*** Cursor right ***
```

```
61110    GOSUB 62380             ' Move cursor to next
position
61120    RETURN
61130 '*** Cursor left or backspace ***
61140    IF CURCOL%=LO.SS%(F.SS%,2) THEN 61210       'Goto
prior field
61150    IF TY.SS$(F.SS%)="N" AND INSTR(" +-
",MID$(VL.SS$(F.SS%),A.SS%,1))<>0 THEN RETURN
61160    CURCOL%=CURCOL%-1:A.SS%=A.SS%-1
'Pos of char in field
61170    IF INSTR("ULX#89",MID$(PIC.SS$(F.SS%),A.SS%,1))=0
THEN 61140 'Spec char?
61180    LOCATE LO.SS%(F.SS%,1),CURCOL%,1
61190    RETURN
61200 '*** Cursor up ***      Move to next field left
61210    GOSUB 62500                      ' Edit check field
data before leaving
61220    IF ERR.MSG% THEN RETURN          ' Error found
61225    EXFLD.SS%=-1                     ' Set Flag to Exit
this Field
61230    IF F.SS%>1 THEN F.SS%=F.SS%-1 ELSE
F.SS%=NUMFLDS.SS%                                 ' Goto
prior fld
61240    RETURN
61250 '*** Cursor down or carriage return - Advance to next
field ***
61260    GOSUB 62500                   ' Edit check field
data before leaving
61270    IF ERR.MSG% THEN  RETURN      ' Don't leave field
if error was found
61275    EXFLD.SS%=-1                  ' Set Flag to Exit
this Field
61277    IF F.SS%=NUMFLDS.SS% AND
INSTR(EXITCHR.SS$,CHR$(127))<>0 THEN EXSCR.SS%=-1  'Test to
leave screen after last field
61280    IF F.SS%<NUMFLDS.SS% THEN F.SS%=F.SS%+1 ELSE
F.SS%=1                           ' Increment fld num to
next fld
61290    RETURN
61300 ' **** Del key pressed ******
61310 '               ' Start Del routine for Numeric fld on
left of decimal pt
61320 IF TY.SS$(F.SS%)="N" AND A.SS%<DECPOS% THEN
MID$(VL.SS$(F.SS%),1)=" "+LEFT$(VL.SS$(F.SS%),A.SS%-
1)+RIGHT$(VL.SS$(F.SS%),LE.SS%(F.SS%)-A.SS%): GOTO 61420
61330 '               ' Start Del routine for Numeric fld on
right of decimal pt
61340 IF TY.SS$(F.SS%)="N" THEN
MID$(VL.SS$(F.SS%),1)=LEFT$(VL.SS$(F.SS%),A.SS%-
1)+MID$(VL.SS$(F.SS%),A.SS%+1,LE.SS%(F.SS%)-A.SS%)+"0" :
GOTO 61420
```

```
61350 '                ' Start Del routine for fld w/o non-
edit chr
61360 IF SPECCHR.SS%(F.SS%)=0 THEN
MID$(VL.SS$(F.SS%),1)=LEFT$(VL.SS$(F.SS%),A.SS%-
1)+MID$(VL.SS$(F.SS%),A.SS%+1,LE.SS%(F.SS%)-A.SS%)+" " :
GOTO 61420
61370 '
61380 CNT.SS%=0                    ' Start del routine for
fld with non-edit chr
61390 WHILE
INSTR("ULX#89",MID$(PIC.SS$(F.SS%),A.SS%+1+CNT.SS%,1))<>0
AND CNT.SS%<LE.SS%(F.SS%)-A.SS% : CNT.SS%=CNT.SS%+1 :  WEND
' Count until next non-edit chr
61400 VL.SS$(F.SS%)=LEFT$(VL.SS$(F.SS%),A.SS%-
1)+MID$(VL.SS$(F.SS%),A.SS%+1,CNT.SS%)+"
"+RIGHT$(VL.SS$(F.SS%),LE.SS%(F.SS%)-A.SS%-CNT.SS%)    '
New value for field
61410 '
61420 CURCOL%=CURCOL%-1:A.SS%=A.SS%-1:GOSUB 62340:GOSUB
62380 'Print fld; Set cursor
61430 RETURN
61440 ' ***** Ins key pressed ***
61450 '            ' Start Ins routine for numeric field on
leftside of dec pt
61460 IF TY.SS$(F.SS%)="N" AND A.SS%<DECPOS% THEN
MID$(VL.SS$(F.SS%),1)=MID$(VL.SS$(F.SS%),2,A.SS%-
1)+"0"+RIGHT$(VL.SS$(F.SS%),LE.SS%(F.SS%)-A.SS%) : GOTO
61580
61470 '            ' Start Ins routine for numeric field on
rightside of dec pt
61480 IF TY.SS$(F.SS%)="N"  THEN
VL.SS$(F.SS%)=LEFT$(VL.SS$(F.SS%),A.SS%-
1)+"0"+MID$(VL.SS$(F.SS%),A.SS%,LE.SS%(F.SS%)-A.SS%) : GOTO
61580
61490 '            ' Start Ins routine for non-numeric w/o
non-edit characters
61500 IF SPECCHR.SS%(F.SS%)=0 THEN
VL.SS$(F.SS%)=LEFT$(VL.SS$(F.SS%),A.SS%-1)+"
"+MID$(VL.SS$(F.SS%),A.SS%,LE.SS%(F.SS%)-A.SS%) : GOTO
61580
61510 '            ' Start Ins routine for non-numeric with
non-edit characters
61520 NEWVL$=LEFT$(VL.SS$(F.SS%),A.SS%-1)+" " :
NEXTCHR$=MID$(VL.SS$(F.SS%),A.SS%,1)
61530 FOR I%=A.SS%+1 TO LE.SS%(F.SS%)
61540   X.SS$=MID$(PIC.SS$(F.SS%),I%,1): IF
INSTR("ULX#89",X.SS$)=0 THEN NEWVL$=NEWVL$+X.SS$ : GOTO
61570
61550   NEWVL$=NEWVL$+NEXTCHR$ :
NEXTCHR$=MID$(VL.SS$(F.SS%),I%,1)
61560 NEXT I%
```

```
61570
VL.SS$(F.SS%)=NEWVL$+MID$(VL.SS$(F.SS%),I%+1,LE.SS%(F.SS%))
61580 CURCOL%=CURCOL%-1:A.SS%=A.SS%-1:GOSUB 62310:GOSUB
62380 ' Print fld; Set cursor
61590 RETURN
61600 ' ***** END key pressed *****
61610 CURCOL%=LO.SS%(F.SS%,2)+LE.SS%(F.SS%)-1  :
A.SS%=LE.SS%(F.SS%)
61620 WHILE INSTR("ULX#89",MID$(PIC.SS$(F.SS%),A.SS%,1))=0
:A.SS%=A.SS%-1:CURCOL%=CURCOL%-1: WEND       ' Look for
special protected characters
61630 LOCATE LO.SS%(F.SS%,1),CURCOL%,1       ' Starting
position in this field
61640 RETURN
61650 ' **** HOME key pressed ****  put cursor at beginning
of field
61660 A.SS%=1                        'Find cursor position for
this field
61670 WHILE INSTR("ULX#89",MID$(PIC.SS$(F.SS%),A.SS%,1))=0
: A.SS%=A.SS%+1 : WEND     ' Look for special protected
characters
61680 CURCOL%=LO.SS%(F.SS%,2)+A.SS%-1
61690 LOCATE LO.SS%(F.SS%,1),CURCOL%,1 ' Starting position
in this field
61700 RETURN
61710 '
61715 '
***********************************************
61720 '        ****  ROUTINE TO EDIT-TEST CHARACTER ENTRY
****
61730 '
***********************************************
61740 '***Check for special character type conversion***
61750 ON INSTR("NDMY",TY.SS$(F.SS%)) GOTO
61790,62100,62000,61950
61760 ON INSTR("ULX#89",MID$(PIC.SS$(F.SS%),A.SS%,1)) GOTO
61890,61920,62130,61800,62060,62100
61770 PRINT "EDIT PICTURE TYPE
";MID$(PIC.SS$(F.SS%),A.SS%,1);" NOT FOUND" : STOP
61780 '
61790 '*** Numeric values; "."; "-"; "+"; " "
61800 ' NOTE: The decimal point is trapped in the"Accept
input data" routine
61810 IF X.SS$>="0" AND X.SS$<="9" THEN RETURN
61820 IF X.SS$<>"+" AND X.SS$<>"-" AND X.SS$<>" " THEN
61850  ' Is this a + or - sign?
61830 IF TY.SS$(F.SS%)="C" THEN RETURN ' +,-," " allowed
anywhere in C type
61840    IF LEFT$(VL.SS$(F.SS%),A.SS%-1)=SPACE$(A.SS%-1) OR
A.SS%=1 THEN RETURN
```

```
61841 '      " "  "+" and "-" sign only allowed in
beginning of number
61850    MSG.SS$=" Only numeric values can be entered here.
Please re-enter. "
61860    GOSUB 62180 : RETURN        ' Print error message;
Exit
61870 '
61880 '*** Upper case or any other character***
61890 IF ASC(X.SS$)>96 AND ASC(X.SS$)<123 THEN
X.SS$=CHR$(ASC(X.SS$)-32)
61900 GOTO 62130
61910 '*** Lower case or any other character***
61920 IF ASC(X.SS$)>65 AND ASC(X.SS$)<91 THEN
X.SS$=CHR$(ASC(X.SS$)+32)
61930 GOTO 62130
61940 '*** Y/N answer only***
61950 IF X.SS$="Y" OR X.SS$="y" THEN X.SS$="Y" : GOTO 62130
61960 IF X.SS$="N" OR X.SS$="n" THEN X.SS$="N" : GOTO 62130
61970    MSG.SS$=" Only 'Y' or 'N' can be entered here.
Please re-enter. "
61980    GOTO 62180        ' Print error message
61990 '*** M/F answer only***
62000 IF X.SS$="M" OR X.SS$="m" THEN X.SS$="M" : GOTO 62130
62010 IF X.SS$="F" OR X.SS$="f" THEN X.SS$="F" : GOTO 62130
62020    MSG.SS$=" Only 'M' or 'F' can be entered here.
Please re-enter. "
62030    GOTO 62180      ' Print error message
62040 '
62050 ' *** Numeric values only ****
62060 IF (ASC(X.SS$)>47 AND ASC(X.SS$)<58) THEN GOTO 62130
62070    MSG.SS$=" Only numeric values can be entered here.
Please re-enter. "
62080    GOTO 62180        ' Print error message
62090 ' *** Numeric values and " " only ***
62100 IF (ASC(X.SS$)>47 AND ASC(X.SS$)<58) OR X.SS$=" "
THEN GOTO 62130
62110    MSG.SS$=" Only numeric values or blanks can be
entered here. Please re-enter. "
62120    GOTO 62180        ' Print error message
62130 RETURN
62140 '
62150 ' ********************************
62160 ' **** Print Error Messages ******
62170 ' ********************************
62180   ERR.MSG%=-1 : SOUND 500,SD.SS%*1:LOCATE 25,INT(81-
LEN(MSG.SS$))/2,0 :COLOR 0,7:PRINT MSG.SS$;: LOCATE
LO.SS%(F.SS%,1),CURCOL%,1
62190   WHILE INKEY$<>"" : WEND    ' Clear input buffer
after error
62200 RETURN
62210 '
```

```
62220 ' ****************************************
62230 ' **** Add character to current field *****
62240 ' ****************************************
62250 IF TY.SS$(F.SS%)="N" AND NEWNUM% THEN
MID$(VL.SS$(F.SS%),1)=MID$(BLNK.SS$,1,A.SS%-
1)+X.SS$+"."+STRING$(LE.SS%(F.SS%),"0") : NEWNUM%=0 :
RETURN  ' Restart fld - New num
62260 IF TY.SS$(F.SS%)<>"N" OR NUMED.SS%=-1 THEN
MID$(VL.SS$(F.SS%),A.SS%,1)=X.SS$ : RETURN
62270 IF LEFT$(VL.SS$(F.SS%),1)=" " THEN
MID$(VL.SS$(F.SS%),1,A.SS%)=MID$(VL.SS$(F.SS%),2,A.SS%-
1)+X.SS$  ELSE  NUMED.SS%=-1 : GOTO 62260    ' Add
character to left of decimal place
62280 RETURN
62290 '
62300 '*********************************
62310 '*** Print new value of field ***
62320 '*********************************
62330 '
62340 COLOR CL.SS%(F.SS%,1),CL.SS%(F.SS%,2)          ' Set
color for this field
62350 LOCATE LO.SS%(F.SS%,1),LO.SS%(F.SS%,2),0 : PRINT
VL.SS$(F.SS%); ' Print field
62360 RETURN
62365 '
62370 '************************************
62380 '**** Move cursor to new location ****
62390 '************************************
62400 IF TY.SS$(F.SS%)<>"N" OR NUMED.SS%<>0 THEN 62420
62410    IF LEFT$(VL.SS$(F.SS%),1)<>" " THEN NUMED.SS%=-1
ELSE A.SS%=DECPOS%-1:CURCOL%=LO.SS%(F.SS%,2)+A.SS%-1:LOCATE
LO.SS%(F.SS%,1),CURCOL%,1 : RETURN
62415 '
62420 IF A.SS%<LE.SS%(F.SS%) THEN
A.SS%=A.SS%+1:CURCOL%=CURCOL%+1 ELSE GOSUB 61260 : RETURN
' Advance cursor or go to next field
62430     IF INSTR("ULX#89",MID$(PIC.SS$(F.SS%),A.SS%,1))=0
THEN 62420
62440     LOCATE LO.SS%(F.SS%,1),CURCOL%,1
62450 RETURN
62460 '
62470 ' ************************************
62480 ' ***  Edit check final field result ***
62490 ' ************************************
62500 IF TY.SS$(F.SS%)<>"N" THEN 62540    ' Check numeric
input range
62510     IF VAL(VL.SS$(F.SS%))>RG.SS(F.SS%,2) THEN
MSG.SS$=" The maximum value allowed in this field is
"+STR$(RG.SS(F.SS%,2)): GOSUB 62180 : RETURN  'Print Err
Msg
```

```
62520     IF VAL(VL.SS$(F.SS%))<RG.SS(F.SS%,1) THEN
MSG.SS$=" The minimum value allowed in this field is
"+STR$(RG.SS(F.SS%,1)): GOSUB 62180 : RETURN  'Print Err
Msg
62530 '
62540 IF TY.SS$(F.SS%)<>"D" THEN 62720          ' Date edit
check
62550     IF VL.SS$(F.SS%)="  /  /  " THEN 62720
' No check
62560
DT.SS=VAL(RIGHT$(VL.SS$(F.SS%),2)+LEFT$(VL.SS$(F.SS%),2)+MI
D$(VL.SS$(F.SS%),4,2))
62570     IF
INSTR("01,02,03,04,05,06,07,08,09,10,11,12",LEFT$(VL.SS$(F.
SS%),2))<>0 THEN 62590
62580         MSG.SS$="Invalid MONTH in date entered. Please
re-enter.":GOSUB 62180 : RETURN
62590     IF INSTR(MID$(VL.SS$(F.SS%),4,2)," ")=0 AND
VAL(MID$(VL.SS$(F.SS%),4,2))>0 AND
VAL(MID$(VL.SS$(F.SS%),4,2))<32 THEN 62620
62600         MSG.SS$="Invalid DAY in date entered.  Please
re-enter.":GOSUB 62180:RETURN
62610 '
62620     IF RG.SS(F.SS%,1)=0 AND RG.SS(F.SS%,2)=0 THEN
62720  ' No check
62630     IF RG.SS(F.SS%,2)<RG.SS(F.SS%,1) THEN 62680
'Does date cross century?
62640         IF DT.SS>=RG.SS(F.SS%,1) AND
DT.SS<=RG.SS(F.SS%,2) THEN 62720
62650             D1.SS$=STR$(RG.SS(F.SS%,1)) :
D2.SS$=STR$(RG.SS(F.SS%,2))
62660             MSG.SS$="The date should be between
"+MID$(D1.SS$,4,2)+"/"+RIGHT$(D1.SS$,2)+"/"+MID$(D1.SS$,2,2
)+" and
"+MID$(D2.SS$,4,2)+"/"+RIGHT$(D2.SS$,2)+"/"+MID$(D2.SS$,2,2
) :GOSUB 62180 : RETURN
62670 '
62680     IF DT.SS>=RG.SS(F.SS%,2) OR DT.SS<=RG.SS(F.SS%,1)
THEN 62720
62690             D1.SS$=STR$(RG.SS(F.SS%,2)) :
D2.SS$=STR$(RG.SS(F.SS%,1))
62700             MSG.SS$="The date should be between
"+MID$(D1.SS$,4,2)+"/"+RIGHT$(D1.SS$,2)+"/"+MID$(D1.SS$,2,2
)+" and
"+MID$(D2.SS$,4,2)+"/"+RIGHT$(D2.SS$,2)+"/"+MID$(D2.SS$,2,2
) :GOSUB 62180 : RETURN
62710 '
62720 RETURN
62860 ' ************************************************
62870 ' ** Test If Monochrome or Color/Graphics Monitor **
62880 ' ************************************************
```

```
62890 DEF SEG=&H40
62900 MONO.SS=(PEEK(&H10) AND &H30)=&H30
62910 IF MONO.SS THEN SCRNSEG.SS%=&HB000 ELSE
SCRNSEG.SS%=&HB800
62920 RETURN
62930 ' ******************************
62940 ' **  Error Handling Routine  **
62950 ' ******************************
62960 ' This routine is used mostly to turn the screen
display back on if
62970 ' a serious problem (such as the screen image file
not found) occurs.
62980 CLS:OUT &H3D8,&H29        ' Turn screen display BACK
ON (if off).
62990                          'The following checks for disk
file errors
63000 IF NOT (ERR=53 OR ERR=57 OR ERR=66 OR (ERR>70 AND
ERR<77)) THEN 63030
63010    LOCATE 10,1:PRINT"There is a problem finding a
file on the disk.   Error code="+STR$(ERR)
63020    LOCATE 11,1:PRINT"(HINT:Check disk for .SCR
file)":PRINT
63030 ON ERROR GOTO 0       ' Reset BASIC ERROR handling
```

```
10 '********** ROUTE FILE LOADING PROGRAM **********
12 SCREEN 0,1,0,0:CLS:CONTROL=0
15 ON ERROR GOTO 150
20 LOCATE 2,1:PRINT "THE FOLLOWING ROUTES ARE CURRENTLY ON
FILE:":PRINT
30 LOCATE 4,1:FILES "*"
40 LOCATE 23,1:INPUT "ENTER THE FILENAME HERE ==> ",ROUTE$
45 LABEL=1
60 OPEN ROUTE$ FOR INPUT AS #1
65 LOCATE 14,20:PRINT "~W=LOADNOTE,NOWAIT/"
70 INPUT #1, N
80 FOR I=1 TO N:FOR J=1 TO 8:INPUT #1,NAVDATA(I,J):NEXT
J:NEXT I:CLOSE #1
90 DISPLAY$=ROUTE$+".WPT"
100 OPEN DISPLAY$ FOR INPUT AS #1
110 INPUT #1,N
120 FOR I=1 TO N:FOR J=1 TO 3:INPUT #1,WAYPTS(I,J):NEXT
J:NEXT I:CLOSE #1
122 GEODATA$=ROUTE$+".CHA"
124 OPEN GEODATA$ FOR INPUT AS #1
125 INPUT #1,GEOLAT$,GEOLONG$,MVARTYPE$:CLOSE #1
130 PRINT "~C=LAST/":CHAIN "BOTHALF",6800,ALL
150 IF ERR=53 THEN LOCATE 1,1:PRINT "~W=FILEERR/":SCREEN
0,1,3,0:INPUT CONTROL
160 IF ERR=52 THEN LOCATE 1,1:PRINT "~W=NAMEERR/":SCREEN
0,1,3,0:INPUT CONTROL
162 IF ERR=76 THEN LOCATE 1,1:PRINT "~W=NAMEERR/":SCREEN
0,1,3,0:INPUT CONTROL
164 IF ERR=64 THEN LOCATE 1,1:PRINT "~W=NAMEERR/":SCREEN
0,1,3,0:INPUT CONTROL
170 SCREEN 0,1,0,0:IF CONTROL=1 THEN RESUME 10
175 ON ERROR GOTO 0
180 END
```

```
10 '***** PROGRAM TO ZERO NAVDATA AND WAYPTS ARRAYS *****
20 FOR I=1 TO 9:FOR J=1 TO 9:NAVDATA(I,J)=0:NEXT J:NEXT I
30 FOR I=1 TO 9:FOR J=1 TO 3:WAYPTS(I,J)=0:NEXT J:NEXT I
40 TOTALDIST=0:CHAIN "BOTHALF",6663,ALL
50 END
```

APPENDIX B

TACTICAL COMPUTER AIDED MISSION PLANNING SYSTEM USER MANUAL

WHAT IS COMPUTER AIDED TACTICAL PLANNING?

INTRODUCTION

This program was designed to assist pilots with tactical mission planning in three ways:

1. Compute aircraft performance parameters for all phases of the mission, both high and low altitude. This includes display of the maximum range profile and the maximum range performance for a selected altitude, and allows a side by side comparison of maximum range mission requirements.
2. Provide the ability to enter, edit, and store on disk, many different low level routes, each defined by up to a maximum of nine (9) lat/long navigation points. These routes may later be recalled for performance computations. Routes of more than nine points can be planned in one of two ways:
    A. Make two separate runs of the entire program.

B. Following jet log completion, select the option to calculate another route, then enter the route and continue with the program. Low level performance will be computed using previously input aircraft performance parameters and computed performance.

3. Combine the low level performance computations with lat/long route data, and automatically prepare a completed jet log for the low level route.

The program supplants the time consuming requirement to trace through NATOPS charts to obtain aircraft performance data, compute aircraft performance and mission requirements, measure low level route navigation data, and finally compute a jet log.

USES FOR COMPUTER AIDED TACTICAL PLANNING

The primary uses for a computer based mission planning system are to save time and allow a range of tactical options for consideration that would not be possible with manual calculation. Additionally, computed aircraft performance data must be at least as accurate as data obtained from the NATOPS charts, offering these tactical options with complete and consistent accuracy. Accurate fuel management, beginning at the mission

planning phase, results in more training per flight hour and more tactical options available per mission than is presently available with manual calculations. With this program, planners could build a file of low level routes to a target. Current intelligence information could be collected and utilized much longer, with tactical decisions regarding route, altitudes, and target selection made much later and closer to actual launch times. These tactical decisions could be based upon weather considerations, target options, information regarding enemy defenses, or political considerations. Additionally, accurate launch position information could be used to compute more accurate mission fuel requirements. For the training environment, use of this program can provide significantly better fuel management, reduced costs per flight hour, and thus more effective training per flight hour. Tradeoff studies in mission requirements, based on altitude, speed, fuel required, and time, may also be easily conducted.

SYSTEM REQUIREMENTS

1. An IBM PC, PCXT, PCAT, Portable or any true compatible.
2. 256K RAM.
3. A double density disk drive or a hard disk drive.
4. Any 80 column display, color or monochrome.
5. DOS 2.0 or higher.

WARNING: Always run this program with no other memory resident software installed. The only exception to this is running the program from a RAMDISK. The RAMDISK must be

created BEFORE copying this program to the disk and running this program.

RUNNING THE PROGRAM FROM A RAMDISK

1. System RAM of 640K total is required.
2. Create the ramdisk of size at least 360K.
3. Copy the program disk to the ramdisk using the *.* method.
4. On the ramdrive prompt C> type the batch filename to start the program, RAMPLAN.
5. Be sure to file route data using B:filename

HOW DOES THE PROGRAM WORK?

There are four (4) sections to the complete program.

1. The MAXIMUM RANGE COMPUTER section.
2. The TAKEOFF and HIGH ALTITUDE CRUISE section.
3. The LOW LEVEL CRUISE AND JET LOG section.
4. The LAT/LONG COORDINATE EDITOR section.

MAXIMUM RANGE COMPUTER SECTION

This section requires 6 inputs, and is a stand alone program. This means that the program is run completely separate from the other programs, and results from this program are not automatically transfered to any other program. In contrast, all the remaining sections are interconnected.

Following the data inputs, aircraft maximum range performance is computed at the altitude for maximum range, and at an alternate altitude which the user specifies. Mission time, fuel, and speeds are computed for a mission distance the user specifies. Results can be compared side by side. Maximum endurance performance is also presented. This information can be used to select actual mission altitudes, and input to the other sections.

TAKEOFF AND HIGH ALTITUDE CRUISE SECTION

This section requires 11 inputs regarding aircraft and desired operational parameters. Aircraft performance is then computed and displayed. The program is designed to output this performance in a form most easily utilized by pilots. There is no attempt made to "micro-manage" the planning process by completely computerizing the entire navigation route. The program is intended to be tactically utilized, to be useful in any environment which may precede the low level ingress route. Carrier based Navy and Marine Corps aircraft often launch and complete a rather lengthy procedure of rendezvous, tanking, and tactical formation. Much of the high altitude navigation route may be open ocean, thereby negating the usefulness of any high altitude route planner. A feature such as this would complicate the user interaction and result in a less useful program.

Following the data inputs, aircraft performance is computed for the TAKEOFF, CLIMB, CRUISE, and DESCENT mission phases.

LOW LEVEL CRUISE AND JET LOG SECTION

This section embraces three (3) low level operational features.

1. Lat/long coordinate editor.
2. High or Low level performance computations.
3. Completed jet log.

The user may choose to enter lat/long data manually or from a file. With manual entry, latitude and longitude are entered, along with magnetic variation at each navigation point. The route may then be filed. The user may choose to build another route, or continue with performance calculations for the route just entered. Computed high or low level performance is combined with the navigation route information, high or low level, to produce a completed jet log.

LAT/LONG COORDINATE EDITOR

This section, while included in the LOW LEVEL CRUISE AND JET LOG section, may also be selected independently from the low level planning section. Performance computations are not made here, however, distance and heading information are presented for each route computed. This section is primarily used to build a file of high or low level routes. Since ANY navigation route can be constructed and filed, this section is well suited to aid

the planner in the navigation computations required for the high altitude routes.

GETTING STARTED

This section will describe how to start the program on either an IBM PC/XT/AT or a true compatible such as a COMPAQ, AT&T PC 6300, or ZENITH 150 or 248. Each data input/output screen will be discussed and explained. Pay particular attention to the NOTES and WARNINGS which will attempt to prevent the user from becoming frustrated with the program.

STARTING THE PROGRAM

Insert the program disk into drive A:, close the drive door, turn on the monitor, and then the computer power switch. Highlight the type of machine you are using from the menu using the arrow direction keys on the numeric keypad, followed by hitting the ENTER key to make the selection. Then highlight which group of programs you want to run, either the Maximum Range Computer or the Mission Planning programs, followed by hitting the ENTER key to make the selection.

INTRODUCTION TO THE MAXIMUM RANGE COMPUTER

There is no introductory screen for this section. The first screen displayed will ask for data entry. The second

screen displayed will contain all of the results of the computations.

WHAT YOU NEED TO GET STARTED PLANNING

The following information will be asked for in the data input screen, and therefore should be available initially upon starting the program.

* Gross Weight At Takeoff
* Drag count for Mach Number 0.6
* Desired Altitude for comparison to the optimum altitude for maximum range.
* A Cruise Leg Distance
* Takeoff Temperature in Degrees Fahrenheit
* Cruise Headwind or Tailwind

DATA OUTPUT SCREEN

There is only one Data Output Screen containing the following information based on the input data:

Screen #1 - Computed Performance at the optimum altitude for maximum range:
* Optimum Cruise Altitude
* Max Range Mach Number
* Specific Range in lb/nm
* Cruise Leg Time
* Cruise Leg Fuel
* True Airspeed and Ground Speed

- Computed Performance for maximum range possible at the desired altitude:
* Optimum Cruise Altitude
* Max Range Mach Number
* Specific Range in lb/nm
* Cruise Leg Time
* Cruise Leg Fuel
* True Airspeed and Ground Speed

- Computed performance at the altitude for maximum endurance:
  * Optimum Endurance Altitude
  * Optimum Endurance Mach Number
  * Optimum Endurance Fuel Flow in lb/hr

NOTE: These optimum endurance numbers are computed for a relatively heavy gross weight. The computed performance is therefore tactically useful in determining the optimum rendezvous and join up altitude.

INTRODUCTION

This is a series of screens to explain how the program is structured. For new users, read the introduction, and select option 1: "output to the screen only", at the end of the introduction. Experienced users should select option 2 or option 3.

WHAT YOU NEED TO GET STARTED PLANNING

The following information will be asked for in the first data input screen, and therefore should be available initially upon starting the program.

* Runway Takeoff Temperature in Degrees Fahrenheit.
* Runway Pressure Altitude
* Takeoff Gross Weight
* Runway Length
* Drag Count for Mach Numbers 0.6, 0.7, 0.8, 0.9.
* Desired Cruise Mach Number
* Desired Cruise Altitude
* Headwind or Tailwind
* Altitude at Start of Descent in MSL
* Altitude at End of Descent in MSL
* Takeoff Fuel Load

DATA INPUT SCREEN

Enter the above data for your flight by typing in a value, then pressing the ENTER key, repeating this for all the values. Default values are displayed, and you may select these by simply putting the cursor on the value, and pressing the ENTER key. Drag count data may be found in NATOPS section 11, or in Appendix A of this user guide. To use Appendix A, compute the drag count for mach number 0.6 only from the NATOPS tables. Then proceed to Appendix A and simply read the drag count for the mach numbers 0.7 through 0.9. Input these drag counts to the Data Input Screen.

DATA OUTPUT SCREENS

There are five (5) Data Output Screens containing the following information based on the input data:

Screen #1 - Takeoff Configuration of the aircraft.

Screen #2 - Computed Takeoff Performance with and without double datum:
* Ground Roll Distance
* Takeoff Speed
* Refusal Speed

Screen #3 - Computed Climb Data using the NATOPS maximum range profile approximations:
* Climb Speed
* Climb Mach
* Climb Time
* Climb Fuel Required
* Distance Covered in Climb

Screen #4 - Computed Cruise Data for the input parameters:
* Ambient Temperature
* Fuel Flow
* Specific Range in lb/nm
* Ground Speed
* True Air Speed

Screen #5 - Computed Descent data for the input parameters and for a minimum fuel profile:
* Descent Fuel Used
* Distance Covered
* Descent Time
* Descent Speed

MISSION PLANNING OPTIONS

Select one of the three options:

- LOW LEVEL ROUTE PLANNER -
  * Manually enter a route, edit the route, file the route, compute performance and a jet log for the route.
  * Load a route from a file, and compute performance and a jet log for the route. NOTE: CAN NOT EDIT ROUTES LOADED FROM A FILE.

- ENTER ONLY LAT/LONG DATA -
  * Manually enter, edit, and file a route.
  * May repeat this procedure for multiple route entry and filing.
  * Distance and Heading information is presented.
  * NOTE: NO AIRCRAFT PERFORMANCE OR JET LOG IS COMPUTED.

- EXIT TO DOS -
  * NOTE: BE SURE TO SAVE THE ROUTE BY FILING IT BEFORE YOU EXIT THE PROGRAM!

LOW LEVEL ROUTE PLANNER

This section will be discussed here in more detail, and will include discussions of the other options in this section.

Screen #1 - Low level data input form:

* Type in a value, followed by pressing the ENTER key. Three values are requested.

    1. Low Level Altitude in MSL
    2. Low Level Speed (Kts. Ground Speed)
    3. Beginning Fuel State for the Route

* WARNING: BE SURE TO ENTER LOW LEVEL ALTITUDE AS MSL!
* NOTE: The Low Level Altitude may be as high as 45000 feet MSL. Therefore, typical cruise altitudes may be entered. When combined with a high altitude navigation route entered in the next section, a cruise leg jet log may be produced. Simply run the Mission Planning program again to compute the low level route portion.

Screen #2 - This shows the format for lat/long data entry as DEG.MIN.SEC and requests the following:
* North or South Latitude (N or S)
* East or West Longitude (E or W)
* East or West Magnetic Variation (E or W)
* Number of points to enter (2 to 9)

Screen #3 - Lat/Long data input uses the following sequence:
* Type the value of the latitude DEGrees, then press the ENTER key.
* Type the value of the latitude MINutes, then press the ENTER key.
* Type the value of the latitude SEConds, then press the ENTER key.
* Repeat this sequence for the longitude values.
* Type the value of the magnetic variation as simply a number. NO PLUS OR MINUS SIGNS. NO EAST (E) or WEST (W).

- EDITING DATA BEFORE PRESSING THE {ENTER} KEY:
    * Position the cursor over the incorrect value, type the correct value, press the ENTER key.

- EDITING DATA AFTER PRESSING THE {ENTER} KEY:
    * Continue entering the remaining points. Following this, you will be given the opportunity to edit any incorrect points.

- FILING DATA -

* Respond to program requests.
* WARNING: Provide only filenames of 8 characters or less. DO NOT TYPE ANY EXTENSIONS (example: .DAT) WITH ANY FILENAMES.
* See the following Note To Experienced Users for more filing details.

NOTE TO EXPERIENCED USERS

Data is filed according to the fully qualified filename definition, however, in NO CASE SHOULD A USER SPECIFY A FILENAME THAT INCLUDES AN EXTENSION. Data may be filed on the B: drive to a separate data disk. Route files are very small, approximately 1K in size, and there is approximately 20K of disk space available on the program disk which can store about 20 route files. The user may desire to store classified or otherwise sensitive routes on a separate data disk in the B: drive, which then may be afforded the appropriate degree of security, or to store all route files on a spearate data disk. To accomplish this, simply respond to the requests for filenames with a more complete filename. For an example, the user would type B:IR203 in response to the requests, instead of simply IR203. To erase files from the program disk, use the DOS command { ERASE filename.* } and substitute the actual name of the route for the filename.

Screen #4 - Output screen of computed navigation data.

Screen #5 - Computed low level aircraft performance data.

Screen #6 - Completed low level jet log.

FINAL OPTIONS

To compute a return profile, the program will begin at the beginning, with the data input form. All values must be entered, even though there is no takeoff to be made. The user must tolerate the takeoff computations and display, in order to obtain accurate return profile performance. This method actually allows re-entry of important aircraft configuration, weight, drag, and fuel information that most certainly has changed from the ingress parameters. The result is very accurate return profile performance data.

Calculation of another route will use the input data already provided and performance data already computed in completing the navigation data and jet log.

ERRORS

Most errors will be intercepted, allowing the user to repeat some action correctly and continue with the program. Two most common errors may occur:

1. Responding to a request for a letter by typing a number. In this case, the number will be accepted, however, the program will probably fail shortly thereafter. In this case, start the

program again by pressing the CTRL-ALT-DEL keys simultaneously.

2. Responding to a request for a number by typing a letter. In this case, the user will see the following message on the monitor screen:

```
?Redo from start
```

Although this looks bad, there is an easy solution. Simply type the correct response, press the ENTER key, and continue with the program. If the screen display and cursor location do not line up for proper data entry, or the program isn't running correctly, then press the CTRL-ALT-DEL keys simultaneously to start the program over from the beginning.

QUICK START REFERENCE INSTRUCTIONS

Many people would like to start using a program without reading the instructions, or bothering with the details of program operation. This quick reference guide has been prepared for this purpose.

LOADING AND STARTING THE PROGRAM

Insert the program disk into drive A:, close the drive door, turn on the monitor, and then the computer power switch. Highlight the type of machine you are using from the menu using the arrow direction keys on the numeric keypad, followed by hitting the ENTER key to make the selection. Highlight the MAX RANGE COMPUTER, followed by hitting the ENTER key to make the selction. Follow the on screen instructions. After returning to the main program selection menu, highlight the MISSION PLANNING option followed by hitting the Enter key to make the selection. Then hit the ENTER key again to select the "READ A SHORT INTRO TO THE PROGRAM" option from the mission planning menu. Follow the on-screen instructions carefully. Proceed slowly and patiently the first time through, taking time to read each screen fully, and to read the associated help screens that appear as different options are highlighted with the arrow direction keys on the numeric keypad to the right of the keyboard. Decide on

an option, highlight the option, then hit the ENTER key to make the selection.

DATA ENTRY FORMS

Type in the correct value asked for in a block, followed by pressing the ENTER key. Some blocks, when they are filled, will automatically enter the value and place the cursor at the next block, waiting for data to be entered. If this occurs, simply continue entering the data for the block containing the cursor, and go on with the program. This is designed to save time. Repeat for all the blocks.

LAT/LONG DATA ENTRY

Type in a value for the DEGrees, then press the ENTER key.
Type in a value for the MINutes, then press the ENTER key.
Type in a value for the SEConds, then press the ENTER key.
Type in a value for the MAGnetic VARiation, then press the ENTER key.
Repeat for all the data points.

EDITING

- EDITING DATA BEFORE PRESSING THE {ENTER} KEY:
  * Position the cursor over the incorrect value, type the correct value, press the ENTER key.

- EDITING DATA AFTER PRESSING THE {ENTER} KEY:
  * Continue entering the remaining points. Following this, you will be given the opportunity to edit any incorrect points.
- FILING DATA -
  * Respond to program requests.
  * WARNING: Provide only filenames of 8 characters or less. DO NOT TYPE ANY EXTENSIONS (example: .DAT) WITH ANY FILENAMES.

FINAL OPTIONS

To compute a return profile, the program will begin at the beginning, with the data input form. All entries must be made, even though there is no takeoff to be made. The user must tolerate the takeoff computations and display, in order to obtain accurate return profile performance. This method actually allows re-entry of important aircraft configuration, weight, drag, and fuel information that most certainly has changed from the ingress parameters. The result is very accurate return profile performance data.

APPENDIX C

DRAG COUNT REFERENCE TABLE

MACH NUMBER			
0.6	0.7	0.8	0.9
DRAG COUNT			
20	21	30	52
21	22	31	55
22	23	33	58
23	24	34	60
24	25	36	63
25	27	37	66
26	28	38	68
27	29	40	71
28	30	41	73
29	31	43	76
30	32	44	79
31	33	46	81
32	34	47	84
33	35	49	86
34	36	50	89
35	37	52	92
36	38	53	94
37	39	55	97
38	40	56	100
39	41	58	102
40	42	59	105
41	43	61	107
42	45	62	110
43	46	64	113
44	47	65	115
45	48	67	118
46	49	68	121
47	50	70	123
48	51	71	126
49	52	73	128
50	53	74	131

DRAG COUNT REFERENCE TABLE

MACH NUMBER			
0.6	0.7	0.8	0.9
DRAG COUNT			
50	53	74	131
51	54	75	134
52	55	77	136
53	56	78	139
54	57	80	141
55	58	81	144
56	59	83	147
57	60	84	149
58	61	86	152
59	63	87	155
60	64	89	157
61	65	90	160
62	66	92	162
63	67	93	165
64	68	95	168
65	69	96	170
66	70	98	173
67	71	99	176
68	72	101	178
69	73	102	181
70	74	104	183
71	75	105	186
72	76	107	189
73	77	108	191
74	78	110	194
75	80	111	197
76	81	112	199
77	82	114	202
78	83	115	204
79	84	117	207
80	85	118	210

DRAG COUNT REFERENCE TABLE

MACH NUMBER			
0.6	0.7	0.8	0.9
DRAG COUNT			
80	85	118	210
81	86	120	212
82	87	121	215
83	88	123	217
84	89	124	220
85	90	126	223
86	91	127	225
87	92	129	228
88	93	130	231
89	94	132	233
90	95	133	236
91	96	135	238
92	98	136	241
93	99	138	244
94	100	139	246
95	101	141	249
96	102	142	252
97	103	144	254
98	104	145	257
99	105	147	259
100	106	148	262
101	107	149	265
102	108	151	267
103	109	152	270
104	110	154	272
105	111	155	275
106	112	157	278
107	113	158	280
108	114	160	283
109	116	161	286
110	117	163	288

DRAG COUNT REFERENCE TABLE

MACH NUMBER			
0.6	0.7	0.8	0.9
DRAG COUNT			
110	117	163	288
111	118	164	291
112	119	166	293
113	120	167	296
114	121	169	299
115	122	170	301
116	123	172	304
117	124	173	307
118	125	175	309
119	126	176	312
120	127	178	314
121	128	179	317
122	129	181	320
123	130	182	322
124	131	184	325
125	133	185	328
126	134	186	330
127	135	188	333
128	136	189	335
129	137	191	338
130	138	192	341
131	139	194	343
132	140	195	346
133	141	197	348
134	142	198	351
135	143	200	354
136	144	201	356
137	145	203	359
138	146	204	362
139	147	206	364
140	148	207	367

DRAG COUNT REFERENCE TABLE

MACH NUMBER			
0.6	0.7	0.8	0.9
DRAG COUNT			
140	148	207	367
141	149	209	369
142	151	210	372
143	152	212	375
144	153	213	377
145	154	215	380
146	155	216	383
147	156	218	385
148	157	219	388
149	158	221	390
150	159	222	393
151	160	223	396
152	161	225	398
153	162	226	400
154	163	228	400
155	164	229	400
156	165	231	400
157	166	232	400
158	167	234	400
159	169	235	400
160	170	237	400
161	171	238	400
162	172	240	400
163	173	241	400
164	174	243	400
165	175	244	400
166	176	246	400
167	177	247	400
168	178	249	400
169	179	250	400
170	180	252	400

DRAG COUNT REFERENCE TABLE

MACH NUMBER			
0.6	0.7	0.8	0.9
DRAG COUNT			
170	180	252	400
171	181	253	400
172	182	255	400
173	183	256	400
174	184	258	400
175	186	259	400
176	187	260	400
177	188	262	400
178	189	263	400
179	190	265	400
180	191	266	400
181	192	268	400
182	193	269	400
183	194	271	400
184	195	272	400
185	196	274	400
186	197	275	400
187	198	277	400
188	199	278	400
189	200	280	400
190	201	281	400
191	202	283	400
192	204	284	400
193	205	286	400
194	206	287	400
195	207	289	400
196	208	290	400
197	209	292	400
198	210	293	400
199	211	295	400
200	212	296	400

DRAG COUNT REFERENCE TABLE

MACH NUMBER			
0.6	0.7	0.8	0.9
DRAG COUNT			
200	212	296	400
201	213	297	400
202	214	299	400
203	215	300	400
204	216	302	400
205	217	303	400
206	218	305	400
207	219	306	400
208	220	308	400
209	222	309	400
210	223	311	400
211	224	312	400
212	225	314	400
213	226	315	400
214	227	317	400
215	228	318	400
216	229	320	400
217	230	321	400
218	231	323	400
219	232	324	400
220	233	326	400
221	234	327	400
222	235	329	400
223	236	330	400
224	237	332	400
225	239	333	400
226	240	334	400
227	241	336	400
228	242	337	400
229	243	339	400
230	244	340	400

DRAG COUNT REFERENCE TABLE

MACH NUMBER			
0.6	0.7	0.8	0.9
DRAG COUNT			
230	244	340	400
231	245	342	400
232	246	343	400
233	247	345	400
234	248	346	400
235	249	348	400
236	250	349	400
237	251	351	400
238	252	352	400
239	253	354	400
240	254	355	400
241	255	357	400
242	257	358	400
243	258	360	400
244	259	361	400
245	260	363	400
246	261	364	400
247	262	366	400
248	263	367	400
249	264	369	400
250	265	370	400
251	266	371	400
252	267	373	400
253	268	374	400
254	269	376	400
255	270	377	400
256	271	379	400
257	272	380	400
258	273	382	400
259	275	383	400
260	276	385	400

DRAG COUNT REFERENCE TABLE

MACH NUMBER			
0.6	0.7	0.8	0.9
DRAG COUNT			
260	276	385	400
261	277	386	400
262	278	388	400
263	279	389	400
264	280	391	400
265	281	392	400
266	282	394	400
267	283	395	400
268	284	397	400
269	285	398	400
270	286	400	400
271	287	400	400
272	288	400	400
273	289	400	400
274	290	400	400
275	292	400	400
276	293	400	400
277	294	400	400
278	295	400	400
279	296	400	400
280	297	400	400
281	298	400	400
282	299	400	400
283	300	400	400
284	301	400	400
285	302	400	400
286	303	400	400
287	304	400	400
288	305	400	400
289	306	400	400
290	307	400	400

DRAG COUNT REFERENCE TABLE

MACH NUMBER			
0.6	0.7	0.8	0.9
DRAG COUNT			
290	307	400	400
291	308	400	400
292	310	400	400
293	311	400	400
294	312	400	400
295	313	400	400
296	314	400	400
297	315	400	400
298	316	400	400
299	317	400	400
300	318	400	400
301	319	400	400
302	320	400	400
303	321	400	400
304	322	400	400
305	323	400	400
306	324	400	400
307	325	400	400
308	326	400	400
309	328	400	400
310	329	400	400
311	330	400	400
312	331	400	400
313	332	400	400
314	333	400	400
315	334	400	400
316	335	400	400
317	336	400	400
318	337	400	400
319	338	400	400
320	339	400	400

DRAG COUNT REFERENCE TABLE

MACH NUMBER			
0.6	0.7	0.8	0.9
DRAG COUNT			
320	339	400	400
321	340	400	400
322	341	400	400
323	342	400	400
324	343	400	400
325	345	400	400
326	346	400	400
327	347	400	400
328	348	400	400
329	349	400	400
330	350	400	400
331	351	400	400
332	352	400	400
333	353	400	400
334	354	400	400
335	355	400	400
336	356	400	400
337	357	400	400
338	358	400	400
339	359	400	400
340	360	400	400
341	361	400	400
342	363	400	400
343	364	400	400
344	365	400	400
345	366	400	400
346	367	400	400
347	368	400	400
348	369	400	400
349	370	400	400
350	371	400	400

DRAG COUNT REFERENCE TABLE

MACH NUMBER			
0.6	0.7	0.8	0.9
DRAG COUNT			
350	371	400	400
351	372	400	400
352	373	400	400
353	374	400	400
354	375	400	400
355	376	400	400
356	377	400	400
357	378	400	400
358	379	400	400
359	381	400	400
360	382	400	400
361	383	400	400
362	384	400	400
363	385	400	400
364	386	400	400
365	387	400	400
366	388	400	400
367	389	400	400
368	390	400	400
369	391	400	400
370	392	400	400
371	393	400	400
372	394	400	400
373	395	400	400
374	396	400	400
375	398	400	400
376	399	400	400
377	400	400	400
378	400	400	400
379	400	400	400
380	400	400	400

DRAG COUNT REFERENCE TABLE

MACH NUMBER			
0.6	0.7	0.8	0.9
DRAG COUNT			
380	400	400	400
381	400	400	400
382	400	400	400
383	400	400	400
384	400	400	400
385	400	400	400
386	400	400	400
387	400	400	400
388	400	400	400
389	400	400	400
390	400	400	400
391	400	400	400
392	400	400	400
393	400	400	400
394	400	400	400
395	400	400	400
396	400	400	400
397	400	400	400
398	400	400	400
399	400	400	400
400	400	400	400

APPENDIX D

SAMPLE TACTICAL PLANNING PROBLEM

A-7 NATOPS PROBLEM

An international incident has occurred which necessitates the rapid protection and extraction of U.S. military personnel within the strife stricken country.

You are the Commande of an A-7E squadron and have been given the task of planning and conducting the missions in support of the extraction of the U.S. personnel. Given the following information, determine the greatest distance from the target at which you could launch and return to the same take-off point.

1. Fuel Load - 14,000# (2,000# in each aero I D)

2. Ordnance Load

   a. 1 AGM-45 (Skrike) on Sta #1 and Sta #8

   b. 3 MK-20 (Rockeye) on Ter on Sta #2

   c. 1 Aero I D (Full JP5) on Sta #3 and Sta #6

   d. 1 AIM 9D (Sidewinder) on Sta #4 and Sta #5

   e. 3 MK-83 (LDGP) on Ter on Sta #7

3. Basic Aircraft Weight - 21,000#

4. Area temperature at sea level - 30°C

5. Use 300# fuel for Start/Taxi/Catapult

6. Rendezvous at 10,000' and 25 NM - use 800# fuel during rendezvous

7. Descend from cruise altitude to target
   a. Descend to 12,000'
   b. Use 2,700#/Hr fuel flow
   c. Maintain 370 Kts ground speed
   d. Begin descent at 40 NM

8. Enroute winds are forecast
   a. To the target - 40 Kts headwind
   b. Return to ship - 70 Kts tailwind
   c. Descents - no wind

9. Use 2,000# fuel in target area

10. Expend the following ordnance
    a. 2 AGM-45
    b. 3 MK-20
    c. 3 MK-83
    d. Retain all other racks, ordnance, tanks

11. Climb to cruise altitude for return will be from sea level.

12. Plan for Charlie on arrival with 1,000# internal fuel on the ball.

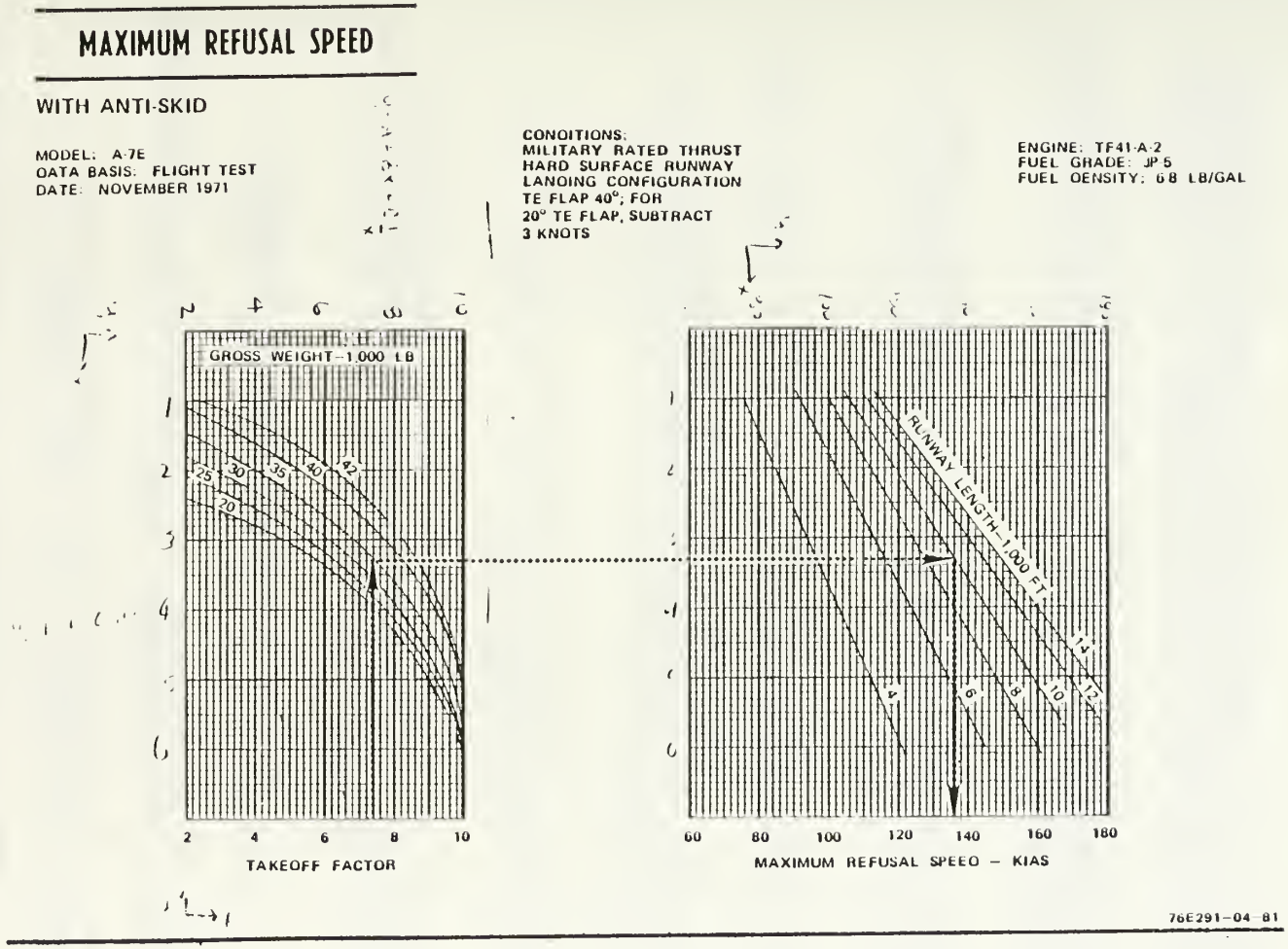

Figure 1

Maximum Refusal Speed

LIST OF REFERENCES

1. Siegel, W.M., Computerization of Tactical Aircraft Performance Data For Fleet Application, M.S. Thesis, Naval Postgraduate School, Monterey, CA, June 1978.

2. Naval Air Development Center Report NADC-81128-60, A-7E Computerized Flight Planning System-Feasibility Investigation, by D.L. Hide and R.C. Polly, pp. 1-17, 31 March 1981.

3. Hill, R.D., The Development of a Performance and Mission Planning Program for the A-7E Aircraft, M.S. Thesis, Naval Postgraduate School, Monterey, CA, September 1984.

4. Rosario, R., Capt. USN (Ret), "To Fly Safely," U.S. Naval Institute Proceedings, pp. 69-73, August 1986.

5. Koger, G.L., The Development and Implementation of Algorithms For An A-7E Performance Calculator, M.S. Thesis, Naval Postgraduate School, Monterey, CA, September 1978.

6. Telephone Conversations between Lt. K.E. Bowersox, Naval Weapons Center, China Lake, CA, and Lt. C.G. Nutter, July 1986.

7. Naval Air Development Center Report NADC-82004-60, Development of Flight Performance Algorithms For The A-7E Aircraft-Final Report, by J. Post and W.E. Mallett, December 1981.

8. Park, C.S., Interactive Microcomputer Graphics, Addison-Wesley Publishing Company, Inc., 1985.

9. Ott, L. and Hildebrand, D.K., Statistical Thinking For Managers, PWS Publishers, 1983.

10. Draper, N.R. and Smith, H., Applied Regression Analysis, John Wiley & Sons, Inc., 1966.

11. Fleet Numerical Oceanography Center FLENUMOCEANCENINST 3710.1, Optimum Path Aircraft Routing System Users Manual, by John P. Garthner, 1 September 1985.

12. Naval Air Development Center NAVAIR 01-45AE-1, Naval Air Training Operational Procedures and Standardization Manual, 1 December 1979.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22304-6145	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, California 93943-5002	2
3.	Department Chairman, Code 67 Department of Aeronautics Naval Postgraduate School Monterey, California 93943-5000	1
4.	Professor D.M. Layton. Code 67Ln Department of Aeronautics Naval Postgraduate School Monterey, California 93943-5000	2
5.	Commanding Officer Naval Strike Warfare Center Fallon, Nevada 89406	1
6.	Commander Light Attack Wing, U.S. Pacific Fleet Naval Air Station Lemoore, California 93245	1
7.	Commander Light Attack Wing 1 Naval Air Station Cecil Field, Florida 32215-0122	1
8.	Commanding Officer AIRTEVRON 5 China Lake, California 93555	1
9.	Lt. Chris Nutter 100 Leidig Circle Monterey, CA 93940	10

220347

Thesis
N972
c.1

Nutter
Development of flight performance algorithms and a tactical computer aided mission planning system for the A-7E aircraft.